Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Table of Contents

; This page intentionally left blank.

```
                                        ;1      .TOC    "REV750.MIC"
                                        ;2      .TOC    "REVISION 98"
                                        ;3      ;       Jeff Peng, J.Heom, Gerard Koeckhoven, Brian Allison, C. E. MCDOWELL
                                        ;4
                                        ;5      .SET/MICROREV=98.
;6      .NOBIN
;8          ;       This revision history indicates which modules have changed, and a SUMMARY of those changes.
;9          ;       Individual revision histories should be consulted for descriptions of the
;10         ;       changes.  The location of each revision history is available from the table
;11         ;       of contents.
;12         ;
;13         ; REV   EXPLAINATION
;14         ;
;15         ; 98    MM      -       Set flags in MM.GET.WRITE.PTE and MM.GET.PTE for TB parity error recovery
;16         ;       IANDF   -       Modify Machine Check routine (uTRAP address 28:) to recovery TB parity error
;17         ;
;18         ; 97    DEFIN   -       Modify VALIDITY check V024 to verify BUS vs MSRC conflict
;19         ;       INSQTI  -       Fix BUS/PROBE.WR and MSRC/TEMP4 hardware conflict.
;20         ;
;21         ; 96    DEFIN   -       Increase the width of micro-word to 81 bits. Add a PATCH field in bit #80.
;22         ;                       Move VSIZE field into bit #84.
;23         ;       PCALL   -       Fix CALLS  and CALLG with write-protected stack area.
;24         ;       OSR     -       Fix instructions PUSHAQ and MOVAQ which increase PC incorrectly.
;25         ;       MPRCHM  -       Jump to PCS to set microrev in SID register.
;26         ;       MISQUE  -       Fix instruction INSQHI with reserved operand.
;27         ;       DECMAL  -       Location 1312 and 1313 remain unused.
;28         ;
;29         ; 95    Replace CCS bread board with PCS.
;30         ;
;31         ; 94    Revision numbers 63 thru 93 were not used to allow the changing as
;32         ;       few roms as possible when changing the rev # for the next change.
;33         ;
;34         ;       MM      -       Save flags and then set MM.NOINT when modifying
;35         ;                       an unaligned operand that is crossing a page boundary.
;36         ;                       Flags are then restored.
;37         ;       MISQUE  -       Save MMTEMP3 (which has the faulting address which aborted the
;38         ;                       instruction) in temp6. This prevents MM from destroying the
;39         ;                       information if a tb-miss occurs when unlocking
;40         ;                       the header. MMTEMP3 is then restored.
;41         ;               -       Additional check provided for the case of a software interlock set and checking of
;42         ;                       bits 1 and 2, if not zero, then reserved operand.
;43         ;       DECMAL  -       Set mm.noint prior to and clearing it afterwards when writing
;44         ;                       data that crosses a page boundary in the general purpose
;45         ;                       write routine for the decimal instructions.
;46         ;               -       some routines were merged to gain some ucode locations in row 5 of ccs
;47         ;       FLOAT   -       use the correct temp (temp3) instead of temp1 in FP IRD1
;48         ;                       specifically for FPA
;49         ;       FILLER  -       remove loc FF7 and FFA for MM fix
;50         ;
;51         ; 93
;52         ;       CHARTS  -       Change some documentation and add a few charts
;53         ;       INIT    -       Fix the 2 changes made in Rev 24
;54         ;                       REassign locations to get return -19 working
;55         ;                       Change BO.INIT_UBA_MAPS to be a subroutine that is called on
```

```
                        both cold and warm boots.
        CONSOL  -       Fix B/number to work without specifying the device.
        INTLOG  -       Remove free location
        FLOAT   -       Fix a POLYD problem that causes the VA to be incorrect, when the arg =0,
                        and the degree not equal to zero and doing a read with FPD set.
                        Recover locations 232 and 233 for use in IANDE.MIC
                        Assign permanent addresses to free locations (where possible)
        VIELD   -       Force some address.
        CONTRL  -       Force some addresses
        CHAR    -       Swap a couple un-constrained words with INIT.MIC to make a return -19
                        work correctly.
        DECMAL  -       Set MM.NOINT during the write of CVTPL to avoid problem with
                        the Write Crossing Page Boundry routine.
        PRLDSV  -       Fix PROBE instructions to flush the XB after restoring the PSL to
                        avoid erroneous ACVs due to prefetch while PSL had modified current
                        mode bits to execute the probe.  Also fix to report no access if
                        probe across a page boundary with first page access ok but not valid,
                        and second page no access.
        IANDE   -       Read modify ACVs go thru read ACV micro trap, and must be
                        detected, so just taking a read tbmiss is no good.
                        Disable interrupts during logging of machine check.
        MM      -       Clear TB to avoid possible conflict between access violation and XB - TB-miss.
        CMODE   -       Cause ODD Address traps when the SP is odd in JSR, RTS, MFPx, MTPx and RTI/RTT
        UVERFY  -       Clean up DSIZE test.
        FILLER  -       Remove words for CMT061.

; 61    DO EVERYTHING
; 60    THIS IS AN INTERIM ECO
        CMODE   -       Fix ODD address check of operands with RTS,JSR, JMP
        INTLOG  -       Recover FREE.00C and FREE.00D, now used in IANDE.
        IANDE   -       More on unaligned read bug.  Was causing KSNV because stack flag was left set if tb miss
                        routine tried to service an interrupt.

; 59    THIS IS AN INTERIM ECO
        CHARTS  -       Documentation change
        MM      -       Fix such that the VA will not be off by 4 when reading across
                        a page boundry and FPD set and you restart the instruction.
        IANDE   -       Reassign FEE  with 3EF
                        Fix unaligned read across a page boundary to not clobber VA if a fault it taken with FPD=1
                        on the second half of the read.
        CHAR    -       Fix SKPC to read longword aligned
                        Change word to long when comparing scr len with dest len
        DECMAL  -       ASHP:    Change byte to word
        CMODE   -       Remove free locations 147c and 147d
        UVERFY  -       Remove free locations 173A and 173B
        FILLER  -       Put 3EF in filler and remove FEE
                        Remove words for CMT059

; 58    THIS IS AN INTERIM ECO
        CHAR    -       Fix problem with R0 in MATCHC
        MM      -       Fix such that the VA will not be off by 8 when writing crossing
                        a page boundery and FPD set and you restart the instruction
        IANDE   -       Fix when FPD set & access violation & no TB miss i.e. the PTE is in the TB but access is not
                        allowed the MDR will not be clobbered.
        FILLER  -       Remove two words for fix in char and mm
```

```
;111
;112     ; 57    THIS IS AN INTERIM ECO
;113     ;       CHAR    -       Optimize MATCHC fix
;114     ;       EDIT    -       Fix EDITPC to get proper data in R5
;115     ;       IANDE   -       Use location 3EE instead of FEE to avoid ROM changes
;116     ;       FILLER  -       Add four words to filler from MATCHC optimization
;117     ;                       SWAP loc FEE with 3EF ,
;118
;119     ; 56    THIS IS AN INTERIM ECO
;120     ;       MM      -       Change location 15FC so that the TB gets cleared. This is
;121     ;                       to avoid a conflict between an access violation and a xb-tb miss.
;122     ;       FLOAT   -       Use locations 232 and 233 for a fix in IANDE but leave them in float
;123     ;                       Fix POLYD and POLYF instructions. Change two locations from word to long.
;124     ;       INTLOG  -       Reassign addresses in EDIV routine, and restore micro-address in
;125     ;                       ADWC routine, which was previously swapped to fix EDIV problem.
;126     ;       CMODE   -       JSR:    Make JSR look for reserved operand fault (register mode)
;127     ;                               at beginning of instruction so PC doesn't get changed.
;128     ;                       TSTB:   Change size[word] to size[idep] when register mode pc
;129     ;       FILLER  -       Remove some microwords
;130
;131     ; 55    THIS IS AN INTERIM ECO
;132     ;       INTLOG  -       Assign permanent addresses to IL.EDIVC and IL.EDIVC1
;133     ;       PCALL   -       The "CALLS and CALLG" instructions have to read the
;134     ;                       mask as a word instead of a longword.
;135     ;       CHAR    -       Fix MATCHC bug fixed in 18
;136     ;       DECMAL  -       Whenever the src len > dest len and the source string is not
;137     ;                       equal to zero and the first byte after a possible zero
;138     ;                       byte is not an illegal ascii character then a
;139     ;                       reserved operand fault will occur.
;140     ;       IANDE   -       When going to WCS using the "XFC" instruction the PC has to
;141     ;                       be backed up and all flags cleared.
;142     ;                       When going to Compatibility Mode check that the first
;143     ;                       PC in CMODE is not ODD.
;144     ;       CMODE   -       DIV:    Zero is an acceptable "expected-negative" result.
;145     ;                               Fix problem that called it overflow and left result unwritten.
;146     ;                       MUL:    Fix bug mentioned in 17
;147     ;                       ASH:    Left shift should not overflow when result sign = original sign.
;148     ;                               Sign extending Q before the test fixes the problem.
;149     ;       FILLER  -       Delete words for rev 55 changes.
;150
;151
;152     ; 54    THIS IS AN INTERIM ECO
;153     ;       OSR -           Code around an undocumented hardware conflict that causes
;154     ;                       MDR_0 and IRDX in the same cycle not to work properly.
;155     ;                       causing QUAD and DOUBLE short literals not to work properly
;156     ;                       when they are the last byte on a page and there
;157     ;                       is a TB MISS on the next page.
;158     ;       DECMAL -        Fix illegal # detection in CVTTP
;159     ;                       In CVTPS instruction the Z-bit was not cleared when source length = 1
;160
;161     ; 54    MISQUE -        Fix problem in QU.UNLOCK routine to restore registers properly when
;162     ;                       faulted out of an instruction using an auto-increment or auto-decrement
;163     ;                       addressing mode.
;164     ;       CMODE -         ASH: Vbit should be cleared if sign of result is
;165     ;                       the same as sign of original operand.
```

```
;166     ;          EDIT -          Fix EDITPC to get proper result in R0, R1
;167     ;          CHAR -          Fix MATCHC bug if obj length > src length
;168     ;          INTLOG -        Swap a micro-address in ADWC routine to fix a bug in EDIV
;169     ;                          Fix EDIV problem such that largest negative number is acceptable.
;170     ;
;171     ;  53      CONSOL -        Fix D P<CR> to not deposit a zero in the PSL.
;172     ;          PRLDSV -        Fix SVPCTX to save the SP in KSP (as well as the PCB) when executed on the kerral stack.
;173     ;          FLOAT -         Fix EMODD to not generate a dirty zero on cases with extreme overflow.
;174     ;          INIT -          Fix invalidating of the TB to not miss the first two locations.
;175     ;          OSR -           Fix OS.BOA.QUAD indexed immediate to return the proper address.
;176     ;          DECMAL -        Fix CVTTP to clear the high nibble of a zero length destination when a page fault occurs trying
;177     ;                          to write the destination.
;178     ;                          Fix CVTLP to properly set/clear Z if overflow occurs.
;179     ;                          Fix CVTPT and CVTPS to properly set CC's for destination length = 0.
;180     ;                          Fix CVTTP and CVTSP to take a reserved operand fault for all illegal bytes that overflow.
;181     ;                          Fix CVTPS and CVTPT to set Z if the srclen > dstlen and dstlen is even.
;182     ;          EDITPC -        Fix REPLACE_SIGN to work if a page fault occurs trying to do the fix up.
;183     ;          IANDE -         Fix machine check code to disable cache on cache parity errors.
;184     ;          MACRO -         New macros for G and H.
;185     ;  52      PCALL -         Fix CALLx to pass the right faulting address on an ACV to memory management.
;186     ;          MM -            Add a label for the fix in PCALL to jump to.
;187     ;          MISQUE -        Fix REMQxI reg dest to properly disable header .ne. addr check
;188     ;          MPRCHM -        Add a BUS/PRB.RD.PTE.K to micro word 0DFA to cure a deadlock problem that
;189     ;                          occurs when running 3 massbuses at the same time
;190     ;  51 ******************** Rev 51 was skipped to save a ROM in an ECO.
;191     ;  50      MISQUE -        Fix interlocked queues to page fault properly.
;192     ;  49      DECMAL -        Fix converts to save/restore the flags when taking an interrupt with FPD set.
;193     ;                          Fix DIVP infinite loop on illegal data bug.
;194     ;          IANDE -         Fix FD opcode decode.
;195     ;                          Remove Q_0 from FX.IE.FD.RET.
;196     ;                          Fix WRT BUS ERR double error halt.  Change halt code to 5.
;197     ;                          Clear FPA traps to avoid erroneous trap.
;198     ;                          Set stack flag on halts to tell console to look at front panel.
;199     ;                          Load PC with SCBB+offset on halt due to vector<1:0>=3.
;200     ;                          Fix disabling of cache on cache machine checks.
;201     ;                          Fix console interrupts to properly clear transmitter interrupts.
;202     ;                          Fix handling of FPA detected faults.
;203     ;                          Fix raising of IPL to 1F on KSNV.
;204     ;
;205     ;  49      FLOAT -         Fix POLYD accuracy and dirty zero bugs.
;206     ;                          Put in rom section of G and H floating.
;207     ;                          Change CRC to clear PSL<C>.
;208     ;                          Change ACBF and ACBD for compatability wiht G and H floating.
;209     ;                          Fix condition codes for CMPD with the FPA.
;210     ;                          Fix size on FPA converts.
;211     ;          GHFLT -         Add G and H floating.
;212     ;          INIT -          Turn off the cache when doing memory test for booting.
;213     ;                          Initialize the UNIBUS in the INIT code.
;214     ;                          Clear TU58 control and status registers in INIT.
;215     ;                          Avoid machine hang clearing TB and CACHE.
;216     ;
;217     ;  49      CONSOL -        Attempt restart for halt codes other than 6.
;218     ;                          Fix clearing of halt pending on restart.
;219     ;          INTLOG -        Fix FPA interface bugs.
;220     ;          PCALL -         Fix CALL and RET to do exact probe if approcimate probe gets TNV.
```

```
;221     ;       MISQUE -        Make change for doing restarts on HALT.
;222     ;                       Fix interlocked queues to set MM.NOINT before trying to clear the lock if reserved operand.
;223     ;       MPRCHM -        Fix deposit to IORESET from the console to not take interrupts.
;224     ;                       Change FPA present IPR to use a branch so it will work from the console.
;225     ;                       Avoid machine hang clearing TB and CACHE.
;226     ;
;227     ;       OSR -           Fix indexed immediate.
;228     ;                       Change write type immediate operands to properly bump the PC.
;229     ;       MM -            Fix bug in MM.PRB.WRITE.SIZ when address crosses a page boundary.
;230     ;                       Fix MM.PB.PROBE to properly restore MDR.
;231     ;                       Fix MM.GET.WRITE.PTE to restore VA for TNV fetching process PTE.
;232     ;                       Avoid machine hang loading the TB.
;233     ;       CHAR -          Fix FPD bug in MATCHC.
;234     ;       PRLDSV -        Avoid machine hang clearing the TB.
;235     ;
;236     ; 48    Initial release to manufacturing.
```

.TOC "Comprehensive Revision History"

	CMT049	CMT050	CMT051	CMT052	CMT053	CMT054	CMT055	CMT056	CMT057	CMT058	CMT059	CMT060	CMT061	CMT062	CMT063	CMT064	CMT065	CMT094	CMT096	CMT097
;CHARTS	016	+	+	+	+	+	+	+	+	+	17	+	+	018	+	+	+	019	+	020
;REGION	010	+	+	+	+	+	+	+	+	+	+	+	+	011	+	+	+	+	+	+
;DEFIN	065	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	066	067
;MACRO	055	+	+	056	057	+	+	+	+	+	+	058	+	+	+	+	+	+	059	+
;INIT	022	+	+	+	023	+	+	+	+	+	024	+	+	+	+	+	+	(025)	+	+
;CONSOL	031	+	+	+	032	+	+	+	+	+	+	+	+	033	+	+	+	+	+	+
;INTLOG	021	+	+	+	+	022	023	+	024	+	+	025	+	+	+	+	+	(026)	+	+
;FLOAT	020	+	+	+	021	+	+	+	022	+	+	+	+	023	+	+	+	(026)	+	+
;VIELD	+	+	+	+	+	+	+	+	+	+	+	+	+	007	+	+	+	+	+	+
;CONTRL	017	+	+	+	+	+	+	+	+	+	+	+	+	018	+	+	+	+	+	+
;PCALL	015	+	+	016	+	+	017	+	+	+	+	+	+	+	+	+	+	+	018	+
;MISQUE	027	028	+	029	+	030	+	+	+	+	+	+	+	031	+	+	+	032	033	034
;CHAR	017	+	+	+	+	018	019	+	020	021	022	+	+	+	+	+	+	(023)	+	+
;DECMAL	021	+	+	+	022	023	+	024	+	+	025	+	+	026	+	+	+	(028)	029	+
;EDIT	010	+	+	+	011	012	+	+	013	+	+	+	+	+	+	+	+	+	+	+
;MPRCHM	035	+	+	036	+	+	+	+	+	+	+	+	037	+	+	+	+	+	038	+
;PRLDSV	017	+	+	+	018	+	+	+	+	+	+	+	+	019	+	+	+	+	+	+
;IANDE	037	+	+	+	038	+	039	040	+	041	042	+	043	044	+	+	+	+	+	+
;MM	023	+	+	024	+	+	+	025	+	026	027	+	+	028	+	+	+	029	+	+
;CMODE	+	+	+	+	+	017	018	019	+	+	+	+	020	021	+	+	+	+	+	+
;OSR	028	+	+	+	029	030	+	+	+	+	+	+	+	(031)	+	+	+	+	031	+
;UVERFY	+	+	+	+	+	+	+	+	+	+	008	+	009	+	+	+	+	(010)	+	+
;FILLER	007	+	+	+	008	009	010	011	012	013	014	+	+	015	+	+	+	016	+	+
;GHROM	0000																	(00.05)		

```
292     .TOC    ''      CMT095 CONTROL STORE PARTS LIST''
293     ; This is in PCS board.
294
295     +-----+   +---------------------------+
296     | xxx |   | Module 'E' number Exxx     |
297     | yyy |   | Micro code revision CMTyyy |      NOTE  *yyy*  IN THE REVISION NUMBER INCICATES NEW PARTS
298     | zzz |   | Part number 23zzzF1-00     |
299     +-----+   +---------------------------+
300
301     +---------+---------+---------+---------+---------+---------+---------+---------+---------+---------+
302     |   108   |   100   |    87   |    79   |    66   |    53   |    40   |    27   |    15   |     5   |
303     |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |
304     |   119   |   116   |   113   |   110   |   107   |   104   |   101   |   098   |   095   |   092   |
305     +---------+---------+---------+---------+---------+---------+---------+---------+---------+---------+
306     |   107   |    99   |    86   |    78   |    65   |    52   |    39   |    26   |    14   |     4   |
307     |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |
308     |   118   |   115   |   112   |   109   |   106   |   103   |   100   |   097   |   094   |   092   |
309     +---------+---------+---------+---------+---------+---------+---------+---------+---------+---------+
310     |   106   |    98   |    85   |    77   |    64   |    51   |    38   |    25   |    13   |     3   |
311     |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |  *096*  |
312     |   119   |   116   |   113   |   110   |   107   |   104   |   101   |   098   |   095   |   092   |
313     +---------+---------+---------+---------+---------+---------+---------+---------+---------+---------+
314      79 - 72   71 - 64   63 - 56   55 - 48   47 - 40   39 - 32   31 - 24   23 - 16   15 - 08   07 - 00
315
316
317
318
319
320
321
322
323
324
325
326
327
328
```

```
.TOC     ''      CMT094 CONTROL STORE PARTS LIST''
```

xxx yyy zzz	Module 'E' number Exxx Micro code revision CMTyyy Part number 23zzzF1-00

NOTE *yyy* IN THE REVISION NUMBER INCICATES NEW PARTS

146 +094+ 967	140 062 894	134 062 890	128 +094+ 963	122 062 884	116 062 880	110 062 877	104 062 874	098 062 871	092 062 867	086 062 861	080 062 858	074 062 854	068 062 851	062 062 847	056 062 844	050 062 843	044 +094+ 946	038 +094+ 941	032 +094+ 936
145 062 899	139 049 762	133 049 757	127 049 753	121 049 750	115 049 748	109 049 746	103 049 744	097 049 741	091 062 866	085 049 734	079 049 732	073 049 727	067 049 724	061 049 721	055 049 719	049 +094+ 949	043 +094+ 945	037 +094+ 940	031 +094+ 935
144 062 898	138 062 893	132 062 889	126 062 886	120 062 883	114 062 879	108 062 876	102 062 873	096 062 870	090 062 865	084 062 860	078 062 857	072 062 853	066 062 850	060 062 846	054 044 507	048 062 842	042 062 836	036 062 830	030 062 824
143 062 897	137 049 760	131 049 755	125 049 752	119 062 882	113 049 747	107 049 745	101 049 743	095 049 740	089 062 864	083 +094+ 955	077 062 856	071 050 778	065 052 786	059 049 720	053 044 506	047 +094+ 948	041 +094+ 944	035 +094+ 939	029 +094+ 934
142 +094+ 966	136 +094+ 965	130 +094+ 964	124 +094+ 962	118 +094+ 961	112 +094+ 960	106 +094+ 959	100 +094+ 958	094 +094+ 957	088 +094+ 956	082 +094+ 954	076 +094+ 953	070 +094+ 952	064 +094+ 951	058 +094+ 950	052 044 505	046 +094+ 947	040 +094+ 943	034 +094+ 938	028 +094+ 933
141 062 895	135 062 891	129 062 888	123 062 885	117 062 881	111 062 878	105 062 875	099 062 872	093 062 868	087 062 862	081 062 859	075 062 855	069 062 852	063 062 848	057 062 845	051 044 504	045 062 839	039 +094+ 942	033 +094+ 937	027 +094+ 932

```
.TOC     ''      CMT063 - CMT093  CONTROL STORE PARTS LIST''
; THE PARTS LISTS FOR CMT063 THROUGH CMT093 ARE NOT INCLUDED SINCE THESE WERE INTERIM ECO'S
```

```
.TOC  ""  CMT062 CONTROL STORE PARTS LIST""
```

xxx yyy zzz
Module 'E' number Exxx Micro code revision CMTyyy Part number 23zzzF1-00

NOTE *yyy* IN THE REVISION NUMBER INCICATES NEW PARTS

146	140	134	128	122	116	110	104	098	092	086	080	074	068	062	056	050	044	038	032
062	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*044*	*062*	*062*	*062*	*062*
900	894	890	887	884	880	877	874	871	867	861	858	854	851	847	844	843	838	832	826
145	139	133	127	121	115	109	103	097	091	085	079	073	067	061	055	049	043	037	031
062	049	049	049	049	049	049	049	049	*062*	049	049	049	049	049	049	049	*062*	*062*	*062*
899	762	757	753	750	748	746	744	741	866	734	732	727	724	721	719	718	837	831	825
144	138	132	126	120	114	108	102	096	090	084	078	072	066	060	054	048	042	036	030
062	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	044	*062*	*062*	*062*	*062*
898	893	889	886	883	879	876	873	870	865	860	857	853	850	846	507	842	836	830	824
143	137	131	125	119	113	107	101	095	089	083	077	071	065	059	053	047	041	035	029
062	049	049	049	*062*	049	049	049	049	*062*	049	*062*	050	052	049	044	*062*	*062*	*062*	*062*
897	760	755	752	882	747	745	743	740	864	733	856	778	786	720	506	841	835	829	823
142	136	130	124	118	112	106	100	094	088	082	076	070	064	058	052	046	040	034	028
062	*062*	049	044	044	044	044	044	*062*	*062*	044	044	044	*062*	044	044	*062*	*062*	*062*	*062*
896	892	754	577	571	565	559	553	869	863	535	529	523	849	511	505	840	834	828	822
141	135	129	123	117	111	105	099	093	087	081	075	069	063	057	051	045	039	033	027
062	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	*062*	044	*062*	*062*	*062*	*062*
895	891	888	885	881	878	875	872	868	862	859	855	852	848	845	504	839	833	827	821

```
.TOC  ""  CMT053 - CMT061 CONTROL STORE PARTS LIST""
; THE PARTS LISTS FOR CMT053 THROUGH CMT061 ARE NOT INCLUDED SINCE THESE WERE INTERIM ECO'S
```

```
.TOC     ''      CMT052 CONTROL STORE PARTS LIST''
```

xxx yyy zzz	Module 'E' number Exxx Micro code revision CMTyyy Part number 23zzzF1-00

NOTE *yyy* IN THE REVISION NUMBER INCICATES NEW PARTS

146 049 769	140 049 763	134 049 758	128 047 581	122 049 751	116 047 569	110 047 563	104 047 557	098 049 742	092 049 739	086 049 735	080 049 770	074 049 728	068 *052* 787	062 049 722	056 044 509	050 047 503	044 049 714	038 049 708	032 049 702
145 049 768	139 049 762	133 049 757	127 049 753	121 049 750	115 049 748	109 049 746	103 049 744	097 049 741	091 049 738	085 049 734	079 049 732	073 049 727	067 049 724	061 049 721	055 049 719	049 049 718	043 049 713	037 049 707	031 049 701
144 049 767	138 049 761	132 049 756	126 047 579	120 047 573	114 047 567	108 047 561	102 047 555	096 044 549	090 047 543	084 047 537	078 049 731	072 047 525	066 049 772	060 044 513	054 044 507	048 46A 501	042 049 712	036 049 706	030 049 700
143 *052* 790	137 049 760	131 049 755	125 049 752	119 049 749	113 049 747	107 049 745	101 049 743	095 049 740	089 049 737	083 049 733	077 *052* 789	071 050 778	065 *052* 786	059 049 720	053 044 506	047 *052* 785	041 *052* 784	035 *052* 783	029 *052* 782
142 049 765	136 049 759	130 049 754	124 044 577	118 044 571	112 044 565	106 044 559	100 044 553	094 044 547	088 044 541	082 044 535	076 044 529	070 044 523	064 044 517	058 044 511	052 044 505	046 049 716	040 049 710	034 049 704	028 049 698
141 049 764	135 044 588	129 046 582	123 046 576	117 046 570	111 046 564	105 046 558	099 046 552	093 046 546	087 049 736	081 046 534	075 049 729	069 046 522	063 049 773	057 044 510	051 044 504	045 049 715	039 049 709	033 049 703	027 049 697

.TOC " CMT050 CONTROL STORE PARTS LIST"

xxx yyy zzz

Module 'E' number Exxx
Micro code revision CMTyyy
Part number 23zz-F1-00

NOTE *yyy* IN THE REVISION NUMBER INCICATES NEW PARTS

146	140	134	128	122	116	110	104	098	092	086	080	074	068	062	056	050	044	038	032
049	049	049	047	049	047	047	047	049	049	049	049	049	049	049	044	047	049	049	049
769	763	758	581	751	569	563	557	742	739	735	770	728	725	722	509	503	714	708	702
145	139	133	127	121	115	109	103	097	091	085	079	073	067	061	055	049	043	037	031
049	049	049	049	049	049	049	049	049	049	049	049	049	049	049	049	049	049	049	049
768	762	757	753	750	748	746	744	741	738	734	732	727	724	721	719	718	713	707	701
144	138	132	126	120	114	108	102	096	090	084	078	072	066	060	054	048	042	036	030
049	049	049	047	047	047	047	047	044	047	047	049	047	049	044	044	46A	049	049	049
767	761	756	579	573	567	561	555	549	543	537	731	525	772	513	507	501	712	706	700
143	137	131	125	119	113	107	101	095	089	083	077	071	065	059	053	047	041	035	029
050	049	049	049	049	049	049	049	049	049	049	*050*	*050*	049	049	044	049	*050*	*050*	*050*
780	760	755	752	749	747	745	743	740	737	733	779	778	723	720	506	717	777	776	775
142	136	130	124	118	112	106	100	094	088	082	076	070	064	058	052	046	040	034	028
049	049	049	044	044	044	044	044	044	044	044	044	044	044	044	044	049	049	049	049
765	759	754	577	571	565	559	553	547	541	535	529	523	517	511	505	716	710	704	698
141	135	129	123	117	111	105	099	093	087	081	075	069	063	057	051	045	039	033	027
049	044	046	046	046	046	046	046	046	049	046	049	046	049	044	044	049	049	049	049
764	588	582	576	570	564	558	552	546	736	534	729	522	773	510	504	715	709	703	697

```
475     .TOC     ''       CMT049 CONTROL STORE PARTS LIST''
476
477       +-----+     +-----------------------------+
478       | xxx |     | Module 'E' number Exxx      |
479       | yyy |     | Micro code revision CMTyyy  |
480       | zzz |     | Part number 23zzzF1-00      |
481       +-----+     +-----------------------------+
482
483       +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
484       | 146 | 140 | 134 | 128 | 122 | 116 | 110 | 104 | 098 | 092 | 086 | 080 | 074 | 068 | 062 | 056 | 050 | 044 | 038 | 032 |
485       |*049*|*049*|*049*| 047 |*049*| 047 | 047 | 047 |*049*|*049*|*049*|*049*|*049*|*049*|*049*| 044 | 047 |*049*|*049*|*049*|
486       | 769 | 763 | 758 | 581 | 751 | 569 | 563 | 557 | 742 | 739 | 735 | 770 | 728 | 725 | 722 | 509 | 503 | 714 | 708 | 702 |
487       +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
488       | 145 | 139 | 133 | 127 | 121 | 115 | 109 | 103 | 097 | 091 | 085 | 079 | 073 | 067 | 061 | 055 | 049 | 043 | 037 | 031 |
489       |*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|
490       | 768 | 762 | 757 | 753 | 750 | 748 | 746 | 744 | 741 | 738 | 734 | 732 | 727 | 724 | 721 | 719 | 718 | 713 | 707 | 701 |
491       +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
492       | 144 | 138 | 132 | 126 | 120 | 114 | 108 | 102 | 096 | 090 | 084 | 078 | 072 | 066 | 060 | 054 | 048 | 042 | 036 | 030 |
493       |*049*|*049*|*049*| 047 | 047 | 047 | 047 | 047 | 044 | 047 | 047 |*049*| 047 |*049*| 044 | 044 | 46A |*049*|*049*|*049*|
494       | 767 | 761 | 756 | 579 | 573 | 567 | 561 | 555 | 549 | 543 | 537 | 731 | 525 | 772 | 513 | 507 | 501 | 712 | 706 | 700 |
495       +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
496       | 143 | 137 | 131 | 125 | 119 | 113 | 107 | 101 | 095 | 089 | 083 | 077 | 071 | 065 | 059 | 053 | 047 | 041 | 035 | 029 |
497       |*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*|*049*| 044 |*049*|*049*|*049*|*049*|
498       | 766 | 760 | 755 | 752 | 749 | 747 | 745 | 743 | 740 | 737 | 733 | 730 | 726 | 723 | 720 | 506 | 717 | 711 | 705 | 699 |
499       +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
500       | 142 | 136 | 130 | 124 | 118 | 112 | 106 | 100 | 094 | 088 | 082 | 076 | 070 | 064 | 058 | 052 | 046 | 040 | 034 | 028 |
501       |*049*|*049*|*049*| 044 | 044 | 044 | 044 | 044 | 044 | 044 | 044 | 044 | 044 | 044 | 044 | 044 |*049*|*049*|*049*|*049*|
502       | 765 | 759 | 754 | 577 | 571 | 565 | 559 | 553 | 547 | 541 | 535 | 529 | 523 | 517 | 511 | 505 | 716 | 710 | 704 | 698 |
503       +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
504       | 141 | 135 | 129 | 123 | 117 | 111 | 105 | 099 | 093 | 087 | 081 | 075 | 069 | 063 | 057 | 051 | 045 | 039 | 033 | 027 |
505       |*049*| 044 | 046 | 046 | 046 | 046 | 046 | 046 | 046 |*049*| 046 |*049*| 046 |*049*| 044 | 044 |*049*|*049*|*049*|*049*|
506       | 764 | 588 | 582 | 576 | 570 | 564 | 558 | 552 | 546 | 736 | 534 | 729 | 522 | 773 | 510 | 504 | 715 | 709 | 703 | 697 |
507       +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

NOTE *yyy* IN THE REVISION NUMBER INCICATES NEW PARTS

```
.TOC     ""     CMT048 CONTROL STORE PARTS LIST""
```

xxx
yyy
zzz

Module 'E' number Exxx
Micro code revision CMTyyy
Part number 23zzzF1-00

NOTE *yyy* IN THE REVISION NUMBER INCICATES NEW PARTS

146	140	134	128	122	116	110	104	098	092	086	080	074	068	062	056	050	044	038	032
047	044	047	047	047	047	047	047	047	047	047	047	047	047	047	044	047	047	047	047
599	593	587	581	575	569	563	557	551	545	539	533	527	521	515	509	503	497	491	485
145	139	133	127	121	115	109	103	097	091	085	079	073	067	061	055	049	043	037	031
47E	045	047	047	047	047	047	047	047	047	047	047	047	047	47E	044	047	047	047	047
598	592	586	580	574	568	562	556	550	544	538	532	526	520	514	508	502	496	490	484
144	138	132	126	120	114	108	102	096	090	084	078	072	066	060	054	048	042	036	030
047	047	045	047	047	047	047	047	044	047	047	047	047	047	044	044	46A	047	47E	47E
597	591	585	579	573	567	561	555	549	543	537	531	525	519	513	507	501	495	489	483
143	137	131	125	119	113	107	101	095	089	083	077	071	065	059	053	047	041	035	029
048	048	047	047	047	047	047	047	047	047	047	048	048	047	047	044	048	048	048	048
596	590	584	578	572	566	560	554	548	542	536	666	665	518	512	506	664	494	488	482
142	136	130	124	118	112	106	100	094	088	082	076	070	064	058	052	046	040	034	028
044	044	044	044	044	044	044	044	044	044	044	044	044	044	044	044	044	044	044	044
595	589	583	577	571	565	559	553	547	541	535	529	523	517	511	505	499	493	487	481
141	135	129	123	117	111	105	099	093	087	081	075	069	063	057	051	045	039	033	027
046	044	046	046	046	046	046	046	046	044	046	046	046	046	044	044	044	044	046	047
594	588	582	576	570	564	558	552	546	540	534	528	522	516	510	504	498	492	486	480

```
.CREF                    ;ENABLE CREF FOR FULL ASSEMBLY
                                ;544      .BIN
```

544; This page intentionally left blank.

```
                                        ;545      .TOC "CHARTS.MIC"
                                        ;546      .TOC "REVISION 20.0"
                                        ;547      ;     Jeff Peng, J.Heom, G.Koeckhoven, C. E. MCDOWELL,  P. R. GUILBAULT
                                        ;548
;549      .NOBIN
;550      .TOC     "      Revision History"
;551      ; 20    Add V025 and V24.25.70 validity-checks to show the hardware conflict
;552      ;        between MSRC/ and BUS/ fields
;553      ; 19    Fill in data for IPR 3F
;554      ; 18    Change some documentation and add a few charts
;555      ; 17    Correct some documentation
;556      ; 16    Add machine check logout and memory control and status registers.
;557      ; 15    Initial release.
```

```
558     .TOC "  Macro Level Charts              : PSL Chart"
559
560
561
562
563         +-----------------------------------------------------------------+
564         |                              PSL                                |
565         +-+-+---+---+---+---+---+---+---------------+---------------+---+
566         |3|3|2 2|2|2|2 2|2 2|2 2|2 1 1 1 1|1 1 1 1 1 1 0 0|0|0|0|0|0 0 0 0|
567         |1|0|9 8|7|6|5 4|3 2|1 0 9 8 7 6|5 4 3 2 1 0 9 8|7|6|5|4|3 2 1 0|
568         +-+-+---+---+---+---+---+---+---------------+---------------+---+
569         |C|T| M |F|I| C | P |M|       |               |D|F|I|T|N|Z|V|C|
570         |M|P| B |P|S| U | R |B|  IPL  |      MBZ      |V|U|V| | | | | |
571         | | | Z |D| | R | V |Z|       |               | | | | | | | | |
572         +-+-+---+---+---+---+---+---+---------------+---------------+---+
573                                       |              PSW              |
574                                       +-------------------------------+
575
576
577                     BITS            DESCRIPTION
578
579                     3:0             Condition Codes
580
581                     4               Trace enable
582
583                     5               Integer Overflow trap enable
584
585                     6               Floating Underflow trap enable
586
587                     7               Decimal Overflow trap enable
588
589                     15:8            Reserved to Digital, must be zero
590
591                     20:16           Interrupt Priority Level
592
593                     21              Reserved to Digital, must be zero
594
595                     22:23           Previous Mode
596
597                     25:24           Current Mode
598
599                     26              Interrupt Stack
600
601                     27              First Part Done
602
603                     29:28           Reserved to Digital, must be zero
604
605                     30              Trace Pending
606
607                     31              Compatibility Mode
```

```
.TOC "  Macro Level Charts              : Internal Processor Registers"
```

COMET IPR'S

		RW				RW	
00	KSP		KERNEL STACK POINTER	20	RXCS		CONSOLE RECEIVE CONTROL/STATUS
01	ESP		EXECUTIVE STACK POINTER	21	RXDB	RO	CONSOLE RECEIVE DATA BUFFER
02	SSP		SUPERVISOR STACK POINTER	22	TXCS		CONSOLE TRANSMIT CONTROL/STATUS
03	USP		USER STACK POINTER	23	TXDB	WO	CONSOLE TRANSMIT DATA BUFFER
04	ISP		INTERRUPT STACK POINTER	24	TBDR		TRANSLATION BUFFER DISABLE REGISTER
05	Reserved			25	CADR		CACHE DISABLE REGISTER
06	Reserved			26	MCESR		MACHINE CHECK ERROR SUMMARY REGISTER
07	Reserved			27	CAER		CACHE ERROR REGISTER
08	P0BR		P0 BASE REGISTER	28	ACCS	RO	ACCELERATOR CONTROL/STATUS REG
09	P0LR		P0 LENGTH REGISTER	29	Reserved		
0A	P1BR		P1 BASE REGISTER	2A	Reserved		
0B	P1LR		P1 LENGTH REGISTER	2B	Reserved		
0C	SBR		SYSTEM BASE REGISTER	2C	Reserved		
0D	SLR		SYSTEM LENGTH REGISTER	2D	Reserved		
0E	Reserved			2E	Reserved		
0F	Reserved			2F	Reserved		
10	PCBB		PROCESS CONTROL BLOCK BASE	30	Reserved		
11	SCBB		SYSTEM CONTROL BLOCK BASE	31	Reserved		
12	IPL		INTERRUPT PRIORITY LEVEL	32	Reserved		
13	ASTR		AST LEVEL REGISTER	33	Reserved		
14	SIRR	WO	SOFTWARE INTERRUPT REQUEST REGISTER	34	Reserved		
15	SISR		SOFTWARE INTERRUPT SUMMARY REGISTER	35	Reserved		
16	Reserved			36	Reserved		
17	CMIERR	RO	CMI ERROR REGISTER	37	IO RESET	WO	INITIALIZE UNIBUS
18	ICCS		INTERVAL CLOCK CONTROL/STATUS	38	MME		MEMORY MANAGEMENT ENABLE
19	NICR	WO	NEXT INTERVAL COUNT REGISTER	39	TBIA	WO	TRANSLATION BUFFER INVALIDATE ALL
1A	ICR	RO	INTERVAL COUNT REGISTER	3A	TBIS	WO	TRANSLATION BUFFER INVALIDATE SINGLE
1B	TODR		TIME OF DAY REGISTER	3B	TB DATA		TRANSLATION BUFFER DATA
1C	CSRS		CONSOLE STORAGE RECEIVER STATUS	3C	Reserved		
1D	CSRD	RO	CONSOLE STORAGE RECEIVER DATA	3D	PMR		PERFORMANCE MONITOR REGISTER
1E	CSTS		CONSOLE STORAGE TRANSMIT STATUS	3E	SID	RO	SYSTEM IDENTIFICATION
1F	CSTD	WO	CONSOLE STORAGE TRANSMIT DATA	3F	TBHP	WO	Probe tb for tb hit

System Identification

33 32 31 30 29 28 27 26	25 24 23 22 21 20 19 18	17 16 15 14 13 12 11 10	09 08 07 06 05 04 03 02
0 0 0 0 0 0 1 0	0 0 0 0 0 0 0 0	Micro Rev	Hardware Rev

```
    .TOC "  Macro Level Charts            : Memory Status & Control Registers"

    ;                               +---------------------------------------------------------------+
    ;                               | CMIERR                  CMI ERROR REGISTER          IPR #^X17|
    ;                               +---------------------------------------------------------------+
    ;                               |3|3|2|2|2|2|2|2|2|2|2|2|1|1|1|1|1|1|1|1|1|1|0|0|0|0|0|0|0|0|0|0|
    ;                               |1|0|9|8|7|6|5|4|3|2|1|0|9|8|7|6|5|4|3|2|1|0|9|8|7|6|5|4|3|2|1|0|
    ;                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ;                               |               | CMIDIS|  SMR  |  RLTO | TBGPR | TBHIT |  BER  |
    ;                               +---------------+-------+-------+-------+-------+-------+-------+
    ;                                                 | | | | |       | | | | |       | | | | |
    ;                               Disable CMI = 1 --+ | | | |       | | | | |       | | | | |
    ;                               Read = 1, Modify = 0 -+ | |       | | | | |       | | | | |
    ;                               Virtual = 0, Physical = 1 -+      | | | | |       | | | | |
    ;                                                 +-------+       | | | | |       | | | | |
    ;                               Saved CPU Mode <1:0> |    |       | | | | |       | | | | |
    ;                                                 +-------+       | | | | |       | | | | |
    ;                                                                 | | | | |       | | | | |
    ;                               Read Lock Timeout = 1 ------------+ | | | |       | | | | |
    ;                               TB G1 Tag Parity Error -------------+ | | |       | | | | |
    ;                               TB G0 Tag Parity Error ---------------+ | |       | | | | |
    ;                               TB G1 Data Parity Error ----------------+ |       | | | | |
    ;                               TB G0 Data Parity Error ------------------+       | | | | |
    ;                                                                                 | | | | |
    ;                               TB Hit on last reference (G0 or G1) --------------+ | | | |
    ;                               Memory Error (Timeout, error) ----------------------+ | | |
    ;                               Uncorrectable Error ----------------------------------+ | |
    ;                               Lost Error ---------------------------------------------+ |
    ;                               Corrected Read Data --------------------------------------+
    ;
    ;
    ;                               +---------------------------------------------------------------+
    ;                               | MCESR      MACHINE CHECK ERROR SUMMARY REGISTER      IPR #^X26|
    ;                               +---------------------------------------------------------------+
    ;                               |3 3 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0|
    ;                               |1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0|
    ;                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    ;                               |                          MBZ                          | | | | |
    ;                               +-------------------------------------------------------+-------+
    ;                                                                                         | | | |
    ;                               BUS ERROR, Refer to Bus Error Register or CMIERR  -------+ | | |
    ;                               TB PARITY ERROR, Refer to CMIERR  -------------------------+ | |
    ;                               MBZ ---------------------------------------------------------+ |
    ;                               XB fetch for ISTRM = 1, Operand Fetch = 0  --------------------+
```

TBDR TRANSLATION BUFFER GROUP DISABLE REGISTER IPR #^X24

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBZ																															

0 = Random Replacement, 1 = Force Replacement
0 = Replace Group 0, 1 = Replace Group 1
1 = Force Miss Group 1
1 = Force Miss Group 0

CADR CACHE DISABLE REGISTER IPR #^X25

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBZ																															

1 = CACHE OFF

CAER CACHE ERROR REGISTER IPR #^X27

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBZ																															

Cache Tag Parity Error
Cache Data Parity Error
Lost Error
Cache Hit

```
744     .TOC "  Macro Level Charts                  : Machine Check Logout Stack"
745     ;                                           MACHINE CHECK EXCEPTION STACK LOGOUT TABLE
746     ;
747     ;                       (SP)      LENGTH PARAMETER     00000028        SUMMARY PARAMETER
748     ;                       (SP)+4    SUMMARY PARAMETER    0000000X
749     ;                       (SP)+8    VA                   XXXXXXXX        1 = CS Parity Error
750     ;                       (SP)+C    PC at time of error  XXXXXXXX        2 = Memory Error,  Cache Parity,
751     ;                       (SP)+10   MDR                  XXXXXXXX            Time Out, or TBUF parity
752     ;                       (SP)+14   SAVED MODE REG       0000000X        6 = Micro-sequencing error,
753     ;                       (SP)+18   RLTO                 0000000X            entered filler micro-code
754     ;                       (SP)+1C   TBGPR                0000000X        7 = Bad IRD
755     ;                       (SP)+20   CAER                 0000000X
756     ;                       (SP)+24   BER                  0000000X
757     ;                       (SP)+28   MCESR                0000000X
758     ;                       (SP)+2C   PC                   XXXXXXXX
759     ;                       (SP)+30   PSL                  XXXXXXXX
```

```
+---------------------------------------------------------------+
| RLTO              READ LOCK TIMEOUT REGISTER         ^X18(SP) |
+---------------------------------------------------------------+
|3 3 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0|
|1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              MBZ                            | |
+-------------------------------------------------------------+-+
                                                               |
Read Lock Time Out on CMI -------------------------------------+


+---------------------------------------------------------------+
| SMR               SAVED CPU MODE REGISTER            ^X14(SP) |
+---------------------------------------------------------------+
|3 3 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0|
|1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              MBZ                      | | | | |
+-------------------------------------------------------+-------+
                                                         | | | |
Read Reference = 1, Modify Reference = 0  ---------------+ | | |
Virtual Reference = 0, Physical Reference = 1 -------------+ | |
Saved CPU Mode <1:0>, 00 = Kernel ---------------------------+ |
                      01 = Exec   -----------------------------+
                      10 = Supervisor
                      11 = User


+---------------------------------------------------------------+
| BER                 BUS ERROR REGISTER               ^X24(SP) |
+---------------------------------------------------------------+
|3 3 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0|
|1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              MBZ                      | | | | |
+-------------------------------------------------------+-------+
                                                         | | | |
Memory Error --------------------------------------------+ | | |
Uncorrectable Error ---------------------------------------+ | |
Lost Error --------------------------------------------------+ |
Corrected Data Error ------------------------------------------+
```

```
;803     .TOC "  Macro Level Charts              : System Control Block"
;804
;805     ;   Vector              Description             IPL             I/E
;806
;807     ;   SCBB+0              Not Used                -               -
;808     ;   SCBB+4              Machine Check           1F              E
;809     ;   SCBB+8              Kernel Stack Invalid    1F              E
;810     ;   SCBB+C              Power Fail              1E              I
;811     ;   SCBB+10             Reserved Opcode         -               E
;812     ;   SCBB+14             Customer Opcode XFC     -               E
;813     ;   SCBB+18             Reserved Operand        -               E
;814     ;   SCBB+1C             Reserved Address Mode   -               E
;815     ;   SCBB+20             Access Violation        -               E
;816     ;   SCBB+24             Translation Invalid     -               E
;817     ;   SCBB+28             Trace Trap              -               E
;818     ;   SCBB+2C             Breakpoint Opcode       -               E
;819     ;   SCBB+30             Compatability Mode      -               E
;820     ;   SCBB+34             Arithmetic Trap         -               E
;821     ;   SCBB+40             CHMK                    -               E
;822     ;   SCBB+44             CHME                    -               E
;823     ;   SCBB+48             CHMS                    -               E
;824     ;   SCBB+4C             CHMU                    -               E
;825     ;   SCBB+54             Corrected Read Data     1A              I
;826     ;   SCBB+60             Write Bus Error         1D              I
;827     ;   SCBB+84             Soft Interrupt          1               I
;828     ;   SCBB+88             Soft Interrupt          2               I
;829     ;   SCBB+8C             Soft Interrupt          3               I
;830     ;   SCBB+90             Soft Interrupt          4               I
;831     ;   SCBB+94             Soft Interrupt          5               I
;832     ;   SCBB+98             Soft Interrupt          6               I
;833     ;   SCBB+9C             Soft Interrupt          7               I
;834     ;   SCBB+A0             Soft Interrupt          8               I
;835     ;   SCBB+A4             Soft Interrupt          9               I
;836     ;   SCBB+A8             Soft Interrupt          A               I
;837     ;   SCBB+AC             Soft Interrupt          B               I
;838     ;   SCBB+AD             Soft Interrupt          C               I
;839     ;   SCBB+B4             Soft Interrupt          D               I
;840     ;   SCBB+B8             Soft Interrupt          E               I
;841     ;   SCBB+BC             Soft Interrupt          F               I
;842     ;   SCBB+C0             Interval Timer          18              I
;843     ;   SCBB+F0             TU-58   Receive         14              I
;844     ;   SCBB+F4             TU-58 Transmit          14              I
;845     ;   SCBB+F8             Console Receive         14              I
;846     ;   SCBB+FC             Console Transmit        14              I
;847     ;   SCBB+110-1?8        MASSBUS                 14-17           I
;848     ;   SCBB+200            Unibus                  14-17           I
;849     ;
;850     ;   MACRO VECTOR BITS<1:0> = 0 ; Process Interrupt on Kernel Stack unless PSL <IS> = 1
;851     ;   MACRO VECTOR BITS<1:0> = 1 ; Process Interrupt or Exception on Interrupt Stack
;852     ;   MACRO VECTOR BITS<1:0> = 2 ; Trap to WCS location 2001 for service if WCS is present .
;853     ;   MACRO VECTOR BITS<1:0> = 3 ; Halt Processor
```

```
;854    .TOC "  Macro Level Charts              : Massbus and Unibus Vector Generation"
;855
;856    ; MASSBUS VECTORS
;857    ;     MDR after CMI WRITE VECTOR from a Massbus Adaptor
;858    ;
;859    ;     +-----------------------------------------------------------------+
;860    ;     |3 3 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0|   XX = The encoded priority plug BR. XX = 00 for BR4
;861    ;     |1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0|                                = 01 for BR5
;862    ;     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                                = 10 for BR6
;863    ;     |                   MBZ                         |0 0 1 X X 0 1 Y Y 0 0|                                = 11 for BR7
;864    ;     +-----------------------------------------------+-+-+-+-+-+-+-+-+-+-+-+   YY = The MBA address jumper selection.
;865    ;                                                                                                  YY = 00 for MBA 0
;866    ;                                                                                                     = 01 for MBA 1
;867    ;                                                                                                     = 10 for MBA 2
;868    ; UNIBUS VECTORS
;869    ;     MDR after CMI WRITE VECTOR from UBI 0
;870    ;     +-----------------------------------------------+-----------------+
;871    ;     |                   MBZ                         |0 1 Z Z Z Z Z Z Z Z|   ZZZZZZZZ = The Unibus vector
;872    ;     +-----------------------------------------------+-+-+-+-+-+-+-+-+-+
;873    ;
;874    ;     MDR after CMI WRITE VECTOR from UBI 1
;875    ;     +-----------------------------------------------+-----------------+
;876    ;     |                   MBZ                         |1 0 Z Z Z Z Z Z Z Z|
;877    ;     +-----------------------------------------------+-+-+-+-+-+-+-+-+-+
```

```
;878     .TOC "  Macro Level charts              : Instruction Name vs Op Code"
;879
;880
;881
;882
;883
;884     ;  ACBB    9D      BGTR    14      CASEW   AF      CVTLP   F9      FFS     EA      MOVQ    7D      resvrd  59
;885     ;  ACBD    6F      BGTRU   1A      CHME    BD      CVTLW   F7      HALT    00      MOVTC   2E      resvrd  5A
;886     ;  ACBF    4F      BICB2   8A      CHMK    BC      CVTPL   36      INCB    96      MOVTUC  2F      resvrd  5B
;887     ;  ACBL    F1      BICB3   8B      CHMS    BE      CVTTP   26      INCL    D6      MOVW    B0      resvrd  77
;888     ;  ACBW    3D      BICL2   CA      CHMU    BF      CVTPT   24      INCW    B6      MOVZBL  9A      RET     04
;889     ;  ADAWI   58      BICL3   CB      CLRB    94      CVTPS   08      INDEX   0A      MOVZBW  9B      ROTL    9C
;890     ;  ADDB2   80      BICPSW  B9      CLRD    7C      CVTRDL  6B      INSQHI  5C      MOVZWL  3C      RSB     05
;891     ;  ADDB3   81      BICW2   AA      CLRF    D4      CVTRFL  4B      INSQTI  5D      MTPR    DA      SBWC    D9
;892     ;  ADDD2   60      BICW3   AB      CLRL    D4      CVTSP   09      INSQUE  0E      MULB2   84      SCAN    2A
;893     ;  ADDD3   61      BISB2   88      CLRQ    7C      CVTWB   33      INSV    F0      MULB3   85      SCKPC   3B
;894     ;  ADDF2   40      BISB3   89      CLRW    B4      CVTW    6D      JMP     17      MULD2   64      SOBGEQ  F4
;895     ;  ADDF3   41      BISL2   C8      CMPB    91      CVTWF   4D      JSB     16      MULD3   65      SOBGTR  F5
;896     ;  ADDL2   C0      BISL3   C9      CMPC3   29      CVTWL   32      LDPCTX  06      MULF2   44      SPANC   2B
;897     ;  ADDL3   C1      BISPSW  B8      CMPC5   2D      DECB    97      LOCC    3A      MULF3   45      SUBB2   82
;898     ;  ADDP4   20      BISW2   A8      CMPD    71      DECL    D7      MATCHC  39      MULL2   C4      SUBB3   83
;899     ;  ADDP6   21      BISW3   A9      CMPF    51      DECW    B7      MCOMB   92      MULL3   C5      SUBD2   62
;900     ;  ADDW2   A0      BITB    93      CMPL    D1      DIVB2   86      MCOML   D2      MULP    25      SUBD3   63
;901     ;  ADDW3   A1      BITL    D3      CMPP3   35      DIVB3   87      MCOMW   B2      MULW2   A4      SUBF2   42
;902     ;  ADWC    D8      BITW    B3      CMPP4   37      DIVD2   66      MFPR    DB      MULW3   A5      SUBF3   43
;903     ;  AOBLEQ  F3      BLBC    E9      CMPV    EC      DIVD3   67      MNEGB   8E      NOP     01      SUBL2   C2
;904     ;  AOBLSS  F2      BLBS    E8      CMPW    B1      DIVF2   46      MNEGD   72      POLYD   75      SUBL3   C3
;905     ;  ASHL    78      BLEQ    13      CMPZV   ED      DIVF3   47      MNEGF   52      POLYF   55      SUBP4   22
;906     ;  ASHP    F8      BLEQU   1B      CRC     0B      DIVL2   C6      MNEGL   CE      PROBR   BA      SUBP6   23
;907     ;  ASHQ    79      BLSS    19      CVTBD   6C      DIVL3   C7      MNEGW   AE      PROBER  0C      SUBW2   A2
;908     ;  BB      E1      BLSSU   1F      CVTBF   4C      DIVP    27      MOVAB   9E      PROBEW  0D      SUBW3   A3
;909     ;  BBCC    E5      BNEQ    12      CVTBL   98      DIVW2   A6      MOVAD   7E      PUSHAB  9F      SVPCTX  07
;910     ;  BBCCI   E7      BNEQU   12      CVTBW   99      DIVW3   A7      MOVAF   DE      PUSHAD  7F      TSTB    95
;911     ;  BBCS    E3      BPT     03      CVTDB   68      EDITPC  38      MOVAL   DE      PUSHAF  DF      TSTD    73
;912     ;  BBS     E0      BRB     11      CVTDF   76      EDIV    7B      MOVAQ   7E      PUSHAL  DF      TSTF    53
;913     ;  BBSC    E4      BRW     31      CVTDL   6A      EMODD   74      MOVAW   3E      PUSHAQ  7F      TSTL    D5
;914     ;  BBSS    E2      BSBB    10      CVTDW   69      EMODF   54      MOVB    90      PUSHAW  3F      TSTW    B5
;915     ;  BBSSI   E6      BSBW    30      CVTFB   48      EMUL    7A      MOVC3   28      PUSHL   DD      XFC     FC
;916     ;  BCC     1E      BVC     1C      CVTFD   56      ESCD    FD      MOVC5   2C      PUSHR   BB      XORB2   8C
;917     ;  BCS     1F      BVS     1D      CVTFL   4A      ESCE    FE      MOVD    70      REI     02      XORB3   8D
;918     ;  BEQL    13      CALLG   FA      CVTFW   49      ESCF    FF      MOVF    50      REMQHI  5E      XORL2   CC
;919     ;  BEQLU   13      CALLS   FB      CVTLB   F6      EXTV    EE      MOVL    D0      REMQTI  5F      XORL3   CD
;920     ;  BGEQ    1E      CASE    8F      CVTLD   6E      EXTZV   EF      MOVP    34      REMQUE  0F      XORW2   AC
;921     ;  BGEQU   1E      CASEL   CF      CVTLF   4E      FFC     EB      MOVPSL  DC      resvrd  57      XORW3   AD
```

```
;922     .TOC "  Macro Level Charts              : ESCD Instruction Name vs Op Code(Two byte op codes FDxx)"
;923
;924
;925
;926
;927
;928     ;  ACBG   4F      EMODH  74      resvrd 15      resvrd 3C      resvrd 93      resvrd BA      resvrd DF
;929     ;  ACBH   6F      MNEGG  52      resvrd 16      resvrd 3D      resvrd 94      resvrd BB      resvrd E0
;930     ;  ADDG2  40      MNEGH  72      resvrd 17      resvrd 3E      resvrd 95      resvrd BC      resvrd E1
;931     ;  ADDG3  41      MOVAH  7E      resvrd 18      resvrd 3F      resvrd 96      resvrd BD      resvrd E2
;932     ;  ADDH2  60      MOVAO  7E      resvrd 19      resvrd 57      resvrd 97      resvrd BE      resvrd E3
;933     ;  ADDH3  61      MOVG   50      resvrd 1A      resvrd 58      resvrd 9A      resvrd BF      resvrd E4
;934     ;  CLRH   7C      MOVH   70      resvrd 1B      resvrd 59      resvrd 9B      resvrd C0      resvrd E5
;935     ;  CLRO   7C      MOVO   7D      resvrd 1C      resvrd 5A      resvrd 9C      resvrd C1      resvrd E6
;936     ;  CMPG   51      MULG2  44      resvrd 1D      resvrd 5B      resvrd 9D      resvrd C2      resvrd E7
;937     ;  CMPH   71      MULG3  45      resvrd 1E      resvrd 5C      resvrd 9E      resvrd C3      resvrd E8
;938     ;  CVTBG  4C      MULH2  64      resvrd 1F      resvrd 5D      resvrd 9F      resvrd C4      resvrd E9
;939     ;  CVTBH  6C      MULH3  65      resvrd 20      resvrd 5E      resvrd A0      resvrd C5      resvrd EA
;940     ;  CVTDH  32      POLYG  55      resvrd 21      resvrd 5F      resvrd A1      resvrd C6      resvrd EB
;941     ;  CVTFG  99      POLYH  75      resvrd 22      resvrd 77      resvrd A2      resvrd C7      resvrd EC
;942     ;  CVTFH  98      PUSHAH 7F      resvrd 23      resvrd 78      resvrd A3      resvrd C8      resvrd ED
;943     ;  CVTGB  48      PUSHAO 7F      resvrd 24      resvrd 79      resvrd A4      resvrd C9      resvrd EE
;944     ;  CVTGF  33      resvrd 00      resvrd 25      resvrd 7A      resvrd A5      resvrd CA      resvrd EF
;945     ;  CVTGH  56      resvrd 01      resvrd 26      resvrd 7B      resvrd A6      resvrd CB      resvrd F0
;946     ;  CVTGL  4A      resvrd 02      resvrd 27      resvrd 80      resvrd A7      resvrd CC      resvrd F1
;947     ;  CVTGW  49      resvrd 03      resvrd 28      resvrd 81      resvrd A8      resvrd CD      resvrd F2
;948     ;  CVTHB  68      resvrd 04      resvrd 29      resvrd 82      resvrd A9      resvrd CE      resvrd F3
;949     ;  CVTHD  F7      resvrd 05      resvrd 2A      resvrd 83      resvrd AA      resvrd CF      resvrd F4
;950     ;  CVTHF  F6      resvrd 06      resvrd 2B      resvrd 84      resvrd AB      resvrd D0      resvrd F5
;951     ;  CVTHG  76      resvrd 07      resvrd 2C      resvrd 85      resvrd AC      resvrd D1      resvrd F8
;952     ;  CVTHL  6A      resvrd 08      resvrd 2D      resvrd 86      resvrd AD      resvrd D2      resvrd F9
;953     ;  CVTHW  69      resvrd 09      resvrd 2E      resvrd 87      resvrd AE      resvrd D3      resvrd FA
;954     ;  CVTLG  4E      resvrd 0A      resvrd 2F      resvrd 88      resvrd AF      resvrd D4      resvrd FB
;955     ;  CVTLH  6E      resvrd 0B      resvrd 30      resvrd 89      resvrd B0      resvrd D5      resvrd FC
;956     ;  CVTRGL 4B      resvrd 0C      resvrd 31      resvrd 8A      resvrd B1      resvrd D6      resvrd FD
;957     ;  CVTRHL 6B      resvrd 0D      resvrd 34      resvrd 8B      resvrd B2      resvrd D7      resvrd FE
;958     ;  CVTWG  4D      resvrd 0E      resvrd 35      resvrd 8C      resvrd B3      resvrd D8      resvrd FF
;959     ;  CVTWH  6D      resvrd 0F      resvrd 36      resvrd 8D      resvrd B4      resvrd D9      SUBG3  43
;960     ;  DIVG2  46      resvrd 10      resvrd 37      resvrd 8E      resvrd B5      resvrd DA      SUBG2  42
;961     ;  DIVG3  47      resvrd 11      resvrd 38      resvrd 8F      resvrd B6      resvrd DB      SUBH2  62
;962     ;  DIVH2  66      resvrd 12      resvrd 39      resvrd 90      resvrd B7      resvrd DC      SUBH3  63
;963     ;  DIVH3  67      resvrd 13      resvrd 3A      resvrd 91      resvrd B8      resvrd DD      TSTG   53
;964     ;  EMODG  54      resvrd 14      resvrd 3B      resvrd 92      resvrd B9      resvrd DE      TSTH   73
```

```
;965     .TOC "  Macro Level Charts              : Op Code vs Instruction Name"
;966
;967
;968
;969
;970
;971     ;   00 HALT      22 SUBP4     48 CVTFB     6E CVTLD     91 CMPB      B7 DECW      DC MOVPSL
;972     ;   01 NOP       23 SUBP6     49 CVTFW     6F ACBD      92 MCOMB     B8 BISPSW    DD PUSHL
;973     ;   02 REI       24 CVTPT     4A CVTFL     70 MOVD      93 BITB      B9 BICPSW    DE MOVAF
;974     ;   03 BPT       25 MULP      4B CVTRFL    71 CMPD      94 CLRB      BA POPR      DE MOVAL
;975     ;   04 RET       26 CVTTP     4C CVTBF     72 MNEGD     95 TSTB      BB PUSHR     DF PUSHAF
;976     ;   05 RSB       27 DIVP      4D CVTWF     73 TSTD      96 INCB      BC CHMK      DF PUSHAL
;977     ;   06 LDPCTX    28 MOVC3     4E CVTLF     74 EMODD     97 DECB      BD CHME      E0 BBS
;978     ;   07 SVPCTX    29 CMPC3     4F ACBF      75 POLYD     98 CVTBL     BE CHMS      E1 BBC
;979     ;   08 CVTPS     2A SCANC     50 MOVF      76 CVTDF     99 CVTBW     BF CHMU      E2 BBSS
;980     ;   09 CVTSP     2B SPANC     51 CMPF      77 resvrd    9A MOVZBL    C0 ADDL2     E3 BBCS
;981     ;   0A INDEX     2C MOVC5     52 MNEGF     78 ASHL      9B MOVZBW    C1 ADDL3     E4 BBSC
;982     ;   0B CRC       2D CMPC5     53 TSTF      79 ASHQ      9C ROTL      C2 SUBL2     E5 BBCC
;983     ;   0C PROBER    2E MOVTC     54 EMODF     7A EMUL      9D ACBB      C3 SUBL3     E6 BBSSI
;984     ;   0D PROBEW    2F MOVTUC    55 POLYF     7B EDIV      9E MOVAB     C4 MULL2     E7 BBCCI
;985     ;   0E INSQUE    30 BSBW      56 CVTFD     7C CLRD      9F PUSHAB    C5 MULL3     E8 BLBS
;986     ;   0F REMQUE    31 BRW       57 resvrd    7C CLRQ      A0 ADDW2     C6 DIVL2     E9 BLBC
;987     ;   10 BSBB      32 CVTWL     58 ADAWI     7D MOVQ      A1 ADDW3     C7 DIVL3     EA FFS
;988     ;   11 BRB       33 CVTWB     59 resvrd    7E MOVAD     A2 SUBW2     C8 BISL2     EB FFC
;989     ;   12 BNEQ      34 MOVP      5A resvrd    7E MOVAQ     A3 SUBW3     C9 BISL3     EC CMPV
;990     ;   12 BNEQU     35 CMPP3     5B resvrd    7F PUSHAD    A4 MULW2     CA BICL2     ED CMPZV
;991     ;   13 BEQL      36 CVTPL     5C INSQHI    7F PUSHAQ    A5 MULW3     CB BICL3     EE EXTV
;992     ;   13 BEQLU     37 CMPP4     5D INSQTI    80 ADDB2     A6 DIVW2     CC XORL2     EF EXTZV
;993     ;   14 BGTR      38 EDITPC    5E REMQHI    81 ADDB3     A7 DIVW3     CD XORL3     F0 INSV
;994     ;   15 BLEQ      39 MATCHC    5F REMQTI    82 SUBB2     A8 BISW2     CE MNEGL     F1 ACBL
;995     ;   16 JSB       3A LOCC      60 ADDD2     83 SUBB3     A9 BISW3     CF CASEL     F2 AOBLSS
;996     ;   17 JMP       3B SKPC      61 ADDD3     84 MULB2     AA BICW2     D0 MOVL      F3 AOBLEQ
;997     ;   18 BGEQ      3C MOVZWL    62 SUBD2     85 MULB3     AB BICW3     D1 CMPL      F4 SOBGEQ
;998     ;   19 BLSS      3D ACBW      63 SUBD3     86 DIVB2     AC XORW2     D2 MCOML     F5 SOBGTR
;999     ;   1A BGTRU     3E MOVAW     64 MULD2     87 DIVB3     AD XORW3     D3 BITL      F6 CVTLB
;1000    ;   1B BLEQU     3F PUSHAW    65 MULD3     88 BISB2     AE MNEGW     D4 CLRF      F7 CVTLW
;1001    ;   1C BVC       40 ADDF2     66 DIVD2     89 BISB3     AF CASEW     D4 CLRL      F8 ASHP
;1002    ;   1D BVS       41 ADDF3     67 DIVD3     8A BICB2     B0 MOVW      D5 TSTL      F9 CVTLP
;1003    ;   1E BCC       42 SUBF2     68 CVTDB     8B BICB3     B1 CMPW      D6 INCL      FA CALLG
;1004    ;   1E BGEQU     43 SUBF3     69 CVTDW     8C XORB2     B2 MCOMW     D7 DECL      FB CALLS
;1005    ;   1F BCS       44 MULF2     6A CVTDL     8D XORB3     B3 BITW      D8 ADWC      FC XFC
;1006    ;   1F BLSSU     45 MULF3     6B CVTRDL    8E MNEGB     B4 CLRW      D9 SBWC      FD ESCD
;1007    ;   20 ADDP4     46 DIVF2     6C CVTBD     8F CASEB     B5 TSTW      DA MTPR      FE ESCE
;1008    ;   21 ADDP6     47 DIVF3     6D CVTWD     90 MOVB      B6 INCW      DB MFPR      FF ESCF
```

```
1009    .TOC "  Macro Level Charts              : ESCD Op Code vs Instruction Name(Two byte op codes FDxx)"
1010
1011
1012
1013
1014
1015    ;  00 resvrd    25 resvrd    4A CVTGL     6F ACBH      91 resvrd    B6 resvrd    DB  resvrd
1016    ;  01 resvrd    26 resvrd    4B CVTRGL    70 MOVH      92 resvrd    B7 resvrd    DC  resvrd
1017    ;  02 resvrd    27 resvrd    4C CVTBG     71 CMPH      93 resvrd    B8 resvrd    DD  resvrd
1018    ;  03 resvrd    28 resvrd    4D CVTWG     72 MNEGH     94 resvrd    B9 resvrd    DE  resvrd
1019    ;  04 resvrd    29 resvrd    4E CVTLG     73 TSTH      95 resvrd    BA resvrd    DF  resvrd
1020    ;  05 resvrd    2A resvrd    4F ACBG      74 EMODH     96 resvrd    BB resvrd    E0  resvrd
1021    ;  06 resvrd    2B resvrd    50 MOVG      75 POLYH     97 resvrd    BC resvrd    E1  resvrd
1022    ;  07 resvrd    2C resvrd    51 CMPG      76 CVTHG     98 CVTFH     BD resvrd    E2  resvrd
1023    ;  08 resvrd    2D resvrd    52 MNEGG     77 resvrd    99 CVTFG     BE resvrd    E3  resvrd
1024    ;  09 resvrd    2E resvrd    53 TSTG      78 resvrd    9A resvrd    BF resvrd    E4  resvrd
1025    ;  0A resvrd    2F resvrd    54 EMODG     79 resvrd    9B resvrd    C0 resvrd    E5  resvrd
1026    ;  0B resvrd    30 resvrd    55 POLYG     7A resvrd    9C resvrd    C1 resvrd    E6  resvrd
1027    ;  0C resvrd    31 resvrd    56 CVTGH     7B resvrd    9D resvrd    C2 resvrd    E7  resvrd
1028    ;  0D resvrd    32 CVTDH     57 resvrd    7C CLRH      9E resvrd    C3 resvrd    E8  resvrd
1029    ;  0E resvrd    33 CVTGF     58 resvrd    7C CLRO      9F resvrd    C4 resvrd    E9  resvrd
1030    ;  0F resvrd    34 resvrd    59 resvrd    7D MOVO      A0 resvrd    C5 resvrd    EA  resvrd
1031    ;  10 resvrd    35 resvrd    5A resvrd    7E MOVAH     A1 resvrd    C6 resvrd    EB  resvrd
1032    ;  11 resvrd    36 resvrd    5B resvrd    7E MOVAO     A2 resvrd    C7 resvrd    EC  resvrd
1033    ;  12 resvrd    37 resvrd    5C resvrd    7F PUSHAH    A3 resvrd    C8 resvrd    ED  resvrd
1034    ;  13 resvrd    38 resvrd    5D resvrd    7F PUSHAO    A4 resvrd    C9 resvrd    EE  resvrd
1035    ;  14 resvrd    39 resvrd    5E resvrd    80 resvrd    A5 resvrd    CA resvrd    EF  resvrd
1036    ;  15 resvrd    3A resvrd    5F resvrd    81 resvrd    A6 resvrd    CB resvrd    F0  resvrd
1037    ;  16 resvrd    3B resvrd    60 ADDH2     82 resvrd    A7 resvrd    CC resvrd    F1  resvrd
1038    ;  17 resvrd    3C resvrd    61 ADDH3     83 resvrd    A8 resvrd    CD resvrd    F2  resvrd
1039    ;  18 resvrd    3D resvrd    62 SUBH2     84 resvrd    A9 resvrd    CE resvrd    F3  resvrd
1040    ;  19 resvrd    3E resvrd    63 SUBH3     85 resvrd    AA resvrd    CF resvrd    F4  resvrd
1041    ;  1A resvrd    3F resvrd    64 MULH2     86 resvrd    AB resvrd    D0 resvrd    F5  resvrd
1042    ;  1B resvrd    40 ADDG2     65 MULH3     87 resvrd    AC resvrd    D1 resvrd    F6 CVTHF
1043    ;  1C resvrd    41 ADDG3     66 DIVH2     88 resvrd    AD resvrd    D2 resvrd    F7 CVTHD
1044    ;  1D resvrd    42 SUBG2     67 DIVH3     89 resvrd    AE resvrd    D3 resvrd    F8  resvrd
1045    ;  1E resvrd    43 SUBG3     68 CVTHB     8A resvrd    AF resvrd    D4 resvrd    F9  resvrd
1046    ;  1F resvrd    44 MULG2     69 CVTHW     8B resvrd    B0 resvrd    D5 resvrd    FA  resvrd
1047    ;  20 resvrd    45 MULG3     6A CVTHL     8C resvrd    B1 resvrd    D6 resvrd    FB  resvrd
1048    ;  21 resvrd    46 DIVG2     6B CVTRHL    8D resvrd    B2 resvrd    D7 resvrd    FC  resvrd
1049    ;  22 resvrd    47 DIVG3     6C CVTBH     8E resvrd    B3 resvrd    D8 resvrd    FD  resvrd
1050    ;  23 resvrd    48 CVTGB     6D CVTWH     8F resvrd    B4 resvrd    D9 resvrd    FE  resvrd
1051    ;  24 resvrd    49 CVTGW     6E CVTLH     90 resvrd    B5 resvrd    DA resvrd    FF  resvrd
```

```
;1052  .TOC " Macro Level Charts                    : Compatibility mode opcode chart"
;1053
;1054  +-----------------+---------+-------------------------------------+  +-----------------+---------+-------------------------------------+
;1055  |   16-Bit        | Legal   |          Faulting                   |  |   16-Bit        | Legal   |          Faulting                   |
;1056  |   Opcode        | Instruc.|          Instructions               |  |   Opcode        | Instruc.|          Instructions               |
;1057  +-----------------+---------+-------------------------------------+  +-----------------+---------+-------------------------------------+
;1058  |  000000         |         | HALT   (Rsvrd   inst fault)         |  | 050000-057777   | BIS     |                                     |
;1059  |  000001         |         | WAIT   (Rsvrd   inst fault)         |  | 060000-067777   | ADD     |                                     |
;1060  |  000002         | RTI     |                                     |  | 070000-070777   | MUL     |                                     |
;1061  |  000003         |         | BPT    (BPT     inst fault)         |  | 071000-071777   | DIV     |                                     |
;1062  |  000004         |         | IOT    (IOT     inst fault)         |  | 072000-072777   | ASH     |                                     |
;1063  |  000005         |         | RESET  (Rsvrd   inst fault)         |  | 073000-073777   | ASHC    |                                     |
;1064  |  000006         | RTT     |                                     |  | 074000-074777   | XOR     |                                     |
;1065  |  000007         |         | MFPT   (Rsvrd   inst fault)         |  | 075000-075007   |         | FADD    (Rsvrd inst fault)          |
;1066  |  000010-000077  |         | Unused (Rsvrd   inst fault)         |  | 075010-075017   |         | FSUB    (Rsvrd inst fault)          |
;1067  |  000100-000107  |         | JMP    (Illegal inst fault)         |  | 075020-075027   |         | FMUL    (Rsvrd inst fault)          |
;1068  |  000110-000177  | JMP     |                                     |  | 075030-075037   |         | FDIV    (Rsvrd inst fault)          |
;1069  |  000200-000207  | RTS     |                                     |  | 075040-075777   |         | Unused  (Rsvrd inst fault)          |
;1070  |  000210-000227  |         | Unused (Rsvrd   inst fault)         |  | 076000-076777   |         | XTD INS (Rsvrd inst fault)          |
;1071  |  000230-000237  |         | SPL    (Rsvrd   inst fault)         |  | 077000-077777   | SOB     |                                     |
;1072  |  000240         | NOP     |                                     |  | 100000-100377   | BPL     |                                     |
;1073  |  000241-000257  | CLR CC'S|                                     |  | 100400-100777   | BMI     |                                     |
;1074  |  000260-000277  | SET CC'S|                                     |  | 101000-101377   | BHI     |                                     |
;1075  |  000300-000377  | SWAB    |                                     |  | 101400-101777   | BLOS    |                                     |
;1076  |  000400-000777  | BR      |                                     |  | 102000-102377   | BVC     |                                     |
;1077  |  001000-001377  | BNE     |                                     |  | 102400-102777   | BVS     |                                     |
;1078  |  001400-001777  | BEQ     |                                     |  | 103000-103377   | BCC,BHIS|                                     |
;1079  |  002000-002377  | BGE     |                                     |  | 103400-103777   | BCS,BLO |                                     |
;1080  |  002400-002777  | BLT     |                                     |  | 104000-104377   |         | EMT     (BPT inst fault)            |
;1081  |  003000-003377  | BGT     |                                     |  | 104000-104777   |         | TRAP    (TRAP inst fault)           |
;1082  |  003400-003777  | BLE     |                                     |  | 105000-105077   | CLRB    |                                     |
;1083  |  004x00-004x07  |         | JSR    (Illegal inst fault)         |  | 105100-105177   | COMB    |                                     |
;1084  |  004x10-004x77  | JSR     |                                     |  | 105200-105277   | INCB    |                                     |
;1085  |  005000-005077  | CLR     |                                     |  | 105300-105377   | DECB    |                                     |
;1086  |  005100-005177  | COM     |                                     |  | 105400-105477   | NEGB    |                                     |
;1087  |  005200-005277  | INC     |                                     |  | 105500-105577   | ADCB    |                                     |
;1088  |  005300-005377  | DEC     |                                     |  | 105600-105677   | SBCB    |                                     |
;1089  |  005400-005477  | NEG     |                                     |  | 105700-105777   | TSTB    |                                     |
;1090  |  005500-005577  | ADC     |                                     |  | 106000-106077   | RORB    |                                     |
;1091  |  005600-005677  | SBC     |                                     |  | 106100-106177   | ROLB    |                                     |
;1092  |  005700-005777  | TST     |                                     |  | 106200-106277   | ASRB    |                                     |
;1093  |  006000-006077  | ROR     |                                     |  | 106300-106377   | ASLB    |                                     |
;1094  |  006100-006177  | ROL     |                                     |  | 106400-106477   |         | MTPS    (Rsvrd inst fault)          |
;1095  |  006200-006277  | ASR     |                                     |  | 106500-106577   | MFPD    |                                     |
;1096  |  006300-006377  | ASL     |                                     |  | 106600-106677   | MTPD    |                                     |
;1097  |  006400-006477  |         | MARK   (Rsvrd inst fault)           |  | 106700-106777   |         | MFPS    (Rsvrd inst fault)          |
;1098  |  006500-006577  | MFPI    |                                     |  | 107000-107777   |         | Unused  (Rsvrd inst fault)          |
;1099  |  006600-006677  | MTPI    |                                     |  | 110000-117777   | MOVB    |                                     |
;1100  |  006700-006777  | SXT     |                                     |  | 120000-127777   | CMPB    |                                     |
;1101  |  007000-007077  |         | CSM    (Rsvrd inst fault)           |  | 130000-137777   | BITB    |                                     |
;1102  |  007100-007777  |         | Unused (Rsvrd inst fault)           |  | 140000-147777   | BICB    |                                     |
;1103  |  010000-017777  | MOV     |                                     |  | 150000-157777   | BISB    |                                     |
;1104  |  020000-027777  | CMP     |                                     |  | 160000-167777   | SUB     |                                     |
;1105  |  030000-037777  | BIT     |                                     |  | 170000-177777   |         | FLOAT   (Rsvrd inst fault)          |
;1106  |  040000-047777  | BIC     |                                     |  |                 |         |                                     |
;1107  +-----------------+---------+-------------------------------------+  +-----------------+---------+-------------------------------------+
```

```
;1107   .TOC "  Macro Level Charts                : Operand Specifier Addressing Modes"
;1108
;1109   ; General Register Addressing
;1110   ;
;1111   ;          7     4 3     0
;1112   ;         +-------+-------+
;1113   ;         ! mode  !  reg  !
;1114   ;         +-------+-------+
;1115   ;
;1116   ; Hex  Dec  Name                          Assembler    r m w a v   PC SP  Indexable?
;1117   ;
;1118   ; 0-3  0-3  literal                       S^#literal   y f f f f   -  -       f
;1119   ;  4    4    indexed                       i[Rx]        y y y y y   f  y       f
;1120   ;  5    5    register                      Rn           y y y f y   u  uq      f
;1121   ;  6    6    register deferred             (Rn)         y y y y y   u  y       y
;1122   ;  7    7    autodecrement                 -(Rn)        y y y y y   u  y       ux
;1123   ;  8    8    autoincrement                 (Rn)+        y y y y y   p  y       ux
;1124   ;  9    9    autoincrement deferred        @(R)+        y y y y y   p  y       ux
;1125   ;  A   10    byte displacement             B^D(Rn)      y y y y y   p  y       y
;1126   ;  B   11    byte displacement deferred    @B^D(Rn)     y y y y y   p  y       y
;1127   ;  C   12    word displacement             W^D(Rn)      y y y y y   p  y       y
;1128   ;  D   13    word displacement deferred    @W^D(Rn)     y y y y y   p  y       y
;1129   ;  E   14    longword displacement         L^D(Rn)      y y y y y   p  y       y
;1130   ;  F   15    longword displacement
;1131   ;           deferred                      @L^D(Rn)     y y y y y   p  y       y
;1132   ;
;1133   ; Program Counter Addressing (reg=15)
;1134   ;
;1135   ;          7     4 3 2 1 0
;1136   ;         +-------+-+-+-+-+
;1137   ;         ! mode  !1 1 1 1!
;1138   ;         +-------+-+-+-+-+
;1139   ;
;1140   ; Hex  Dec  Name                          Assembler    r m w a v   PC SP  Indexable?
;1141   ;
;1142   ;  8    8    immediate                     I^#constant  y u u y y   -  -       y
;1143   ;  9    9    absolute                      @#address    y y y y y   -  -       y
;1144   ;  A   10    byte relative                 B^address    y y y y y   -  -       y
;1145   ;  B   11    byte relative deferred        @B^address   y y y y y   -  -       y
;1146   ;  C   12    word relative                 W^address    y y y y y   -  -       y
;1147   ;  D   13    word relative deferred        @W^address   y y y y y   -  -       y
;1148   ;  E   14    long word relative            L^address    y y y y y   -  -       y
;1149   ;  F   15    long word relative
;1150   ;           deferred                      @L^address   y y y y y   -  -       y
;1151   ;
;1152   ;    D -  displacement                      r -  read access
;1153   ;    i -  any indexable addressing mode     m -  modify access
;1154   ;    - -  logically impossible              w -  write access
;1155   ;    f -  reserved addressing mode fault    a -  address access
;1156   ;    p -  Program Counter addressing        v -  field access
;1157   ;    u -  UNPREDICTABLE
;1158   ;    uq - UNPREDICTABLE for quad and double (and field if position + size
;1159   ;         greater than 32)
;1160   ;    ux - UNPREDICTABLE for index register same as base register
```

```
;1161    .TOC "  Macro Level Charts              : Console Commands"
;1162
;1163    ;     E [<QUALIFIER-LIST>] [<SP> <ADDRESS>] <CR>                ; Examine
;1164    ;     D [<QUALIFIER-LIST>] <SP> <ADDRESS> <SP> <DATA> <CR>      ; Deposit
;1165    ;         Qualifiers :    /B        ; Size IS Byte
;1166    ;                         /W        ; Size IS Word
;1167    ;                         /L        ; Size IS Long
;1168    ;                         /V        ; Virtual Address
;1169    ;                         /P        ; Physical Address
;1170    ;                         /I        ; IPR
;1171    ;                         /G        ; GPR
;1172    ;             Address     : Hex Number
;1173    ;                          *        ; Last location
;1174    ;                          P        ; The PSL
;1175    ;
;1176    ;     E<CR>                                                     ; Examine Next Location
;1177    ;
;1178    ;     D<SP> + <SP> <DATA> <SP> <CR>                             ; Deposit Next Location
;1179    ;
;1180    ;     H <CR>                                                    ; HALT (No-op)
;1181    ;
;1182    ;     I <CR>                                                    ; Initialize
;1183    ;
;1184    ;     T <CR>                                                    ; Execute Micro Verify Sequence
;1185    ;
;1186    ;     C <CR>                                                    ; Continue the Processor
;1187    ;
;1188    ;     N <CR>                                                    ; Single Step
;1189    ;
;1190    ;     B [<QUALIFIER-LIST>] [<SP> ddcu ] <CR>                    ; Boot
;1191    ;         Qualifiers : /X           ; Inhibit Execution of Microverify Sequence
;1192    ;                    : /Hex Number  ; Boot Control Flags (R5)
;1193    ;         dd                        ; Device
;1194    ;         c                         ; Adapter
;1195    ;         u                         ; Unit number
;1196    ;
;1197    ;     S <SP> <ADDRESS> <CR>                                     ; Start
;1198    ;                                  ; Initialize the Processor
;1199    ;                                  ; PC <- Address
;1200    ;                                  ; Start the processor
;1201    ;
;1202    ;     X <SP><ADDRESS><SP><0'COUNT><CR><CKSUM1><DATA><CKSUM2>    ; Binary Load
;1203    ;         Address     ; Start Address of Load
;1204    ;         Count       ; No. of Bytes (Unsigned 30 bit number)
;1205    ;         Cksum1      ; 2's Comp Checksum of Command String
;1206    ;         Data        ; Count Bytes of Binary Data
;1207    ;         Cksum2      ; 2's Comp Checksum of Data
;1208    ;
;1209    ;     X <SP> <ADDRESS> <SP> <1'COUNT> <CR> <CKSUM>              ; Binary Read
;1210    ;         Address     ; Start Address of Load
;1211    ;         Count       ; No. of Bytes (Unsigned 30 bit number)
;1212    ;         Cksum       ; 2's Comp Checksum of Command String
```

```
1213    .TOC "  Macro Level Charts              : Console Error and Halt Codes"
1214
1215
1216    ;   +-----------------------------------------------------------------------------+
1217    ;   |                          Console Error Codes                                |
1218    ;   +-----------------------------------------------------------------------------+
1219    ;   |                                                                             |
1220    ;   |    20       ACV, TNV, or Machine Check during a read or write.              |
1221    ;   |                                                                             |
1222    ;   |    11       Error in accessing IPR, or PSL.                                 |
1223    ;   |                                                                             |
1224    ;   |    30       Binary transfer checksum error.                                 |
1225    ;   |                                                                             |
1226    ;   |    33       Unrecognizable Boot device.                                     |
1227    ;   |                                                                             |
1228    ;   |    34       Controller not A, B, C, or D in BOOT command.                   |
1229    ;   |                                                                             |
1230    ;   +-----------------------------------------------------------------------------+
1231
1232
1233
1234
1235
1236    ;   +-----------------------------------------------------------------------------+
1237    ;   |                          Console Halt Codes                                 |
1238    ;   +-----------------------------------------------------------------------------+
1239    ;   |                                                                             |
1240    ;   |    1  Executed TEST console command                                         |
1241    ;   |    2  Control P Halt or single macro instruction mode (ie >>>N<CR>)         |
1242    ;   |    4  Interrupt Stack Not Valid                                             |
1243    ;   |    5  Double Bus Write Error Halt                                           |
1244    ;   |    6  Halt Instruction Executed                                             |
1245    ;   |    7  Vector Bits <1:0>=3, Halt at Vector                                   |
1246    ;   |    8  Vector Bits <1:0>=2, WCS disabled or not present                      |
1247    ;   |    A  Change Mode Instruction executed on Interrupt Stack                   |
1248    ;   |    B  Change Mode Instruction executed and vector <1:0>not=0                |
1249    ;   |    11 Power up and can't find RPB, FPS1 at RESTART/HALT                     |
1250    ;   |    12 Power up and warm start flag false FPS1 at RESTART/HALT               |
1251    ;   |    13 Power up and can't find good 64K of memory                            |
1252    ;   |    14 Power up and booting, but bad Boot ROM or no ROM                      |
1253    ;   |    15 Power up and cold start flag set during boot subroutine               |
1254    ;   |    16 Power up halt FPS1 at HALT position                                   |
1255    ;   |    FF Micro verify test failure                                             |
1256    ;   +-----------------------------------------------------------------------------+
```

```
.TOC "  Macro Level Charts              : Micro Verify Error Codes"
```

COD E	PC+2	TEST NAME/ERROR MESSAGE	COD E	PC+2	TEST NAME/ERROR MESSAGE
'@'		RBUS, WBUS TEST	'O'		XB/IR/OSR BIT TEST
	000	BAD BIT IN DREG OR SUPROT		0F1	ERROR IN XB<31:0>
	001	BAD BIT IN RBUS OR WBUS		0F2	ERROR IN XB<63:32>
				0F4	ERROR IN IR
'C'		MBUS TEST		0F7	ERROR IN OSR
	031	BAD BIT IN QREG			
	032	BAD BIT IN MBUS	'Q'		SOURCE XB PC INCREMENT TEST
				111	ERROR SOURCING ONE BYTE FROM XB
'E'		SCRATCH PAD BIT TEST		112	ERROR SOURCING 2 BYTES FROM XB OR INCREMENTING PC BY 1
	051	ERROR CLEARING RTEMP		114	ERROR SOURCING AN UNALIGNED LONGWORD OR INC PC BY 2
	052	ERROR FILLING RTEMP WITH ONES		117	ERROR INCREMENTING PC BY 4
	054	ERROR CLEARING GPR			
	057	ERROR FILLING GPR WITH ONES	'R'		RNUM/DSIZE TEST
	058	ERROR CLEARING IPR		121	ERROR READING DSIZE ROM OPERAND 1
	05B	ERROR FILLING IPR WITH ONES		122	ERROR LOADING/READING RNUM
	05D	ERROR CLEARING MTEMP		124	ERROR READING DSIZE ROM OPERAND 2
	05E	ERROR FILLING MTEMP WITH ONES		127	ERROR LOADING/READING RNUM
				128	ERROR READING DSIZE ROM OPERAND 3
'F'		MTEMP EXPLICIT ADDRESS TEST		12B	ERROR LOADING/READING RNUM
	061	ERROR ADDRESSING MTEMP0		12D	ERROR READING DSIZE ROM OPERAND 4
	062	ERROR ADDRESSING MTEMP1		12E	ERROR LOADING/READING RNUM
	064	ERROR ADDRESSING MTEMP2			
	067	ERROR ADDRESSING MTEMP4	'T'		RNUM/DSIZE TEST CONTINUED
	068	ERROR ADDRESSING MTEMP8		141	ERROR READING DSIZE ROM OPERAND 5
				142	ERROR LOADING/READING RNUM
'I'		RTEMP EXPLICIT ADDRESS TEST		144	ERROR READING DSIZE ROM OPERAND 6
	091	ERROR ADDRESSING RTEMP0			
	092	ERROR ADDRESSING RTEMP1	'X'		CACHE PARITY ERROR TEST
	094	ERROR ADDRESSING RTEMP2		181	FAILED TO GET CACHE PARITY ERROR
	097	ERROR ADDRESSING RTEMP4		182	BAD MACHINE CHECK ERROR SUMMARY REGISTER
	098	ERROR ADDRESSING RTEMP8		184	BAD CACHE ERROR REGISTER
'J'		IPR EXPLICIT ADDRESS TEST	'['		TB PARITY ERROR TEST
	0A1	ERROR ADDRESSING IPR0		1B1	FAILED TO GET GROUP 0 TB PARITY ERROR
	0A2	ERROR ADDRESSING IPR1		1B2	BAD TB GROUP PARITY ERROR REGISTER
	0A4	ERROR ADDRESSING IPR2		1B4	BAD MACHINE CHECK ERROR SUMMARY REGISTER
	0A7	ERROR ADDRESSING IPR4		1B7	FAILED TO GET GROUP 1 TB PARITY ERROR
	0A8	ERROR ADDRESSING IPR8		1B8	BAD TB GROUP PARITY ERROR REGISTER
				1BB	BAD MACHINE CHECK ERROR SUMMARY REGISTER
'L'		GPR EXPLICIT ADDRESS TEST			
	0C1	ERROR ADDRESSING R0	']'		CONTROL STORE PARITY ERROR TEST
	0C2	ERROR ADDRESSING R1		1D1	FAILED TO GET CONTROL STORE PARITY ERROR
	0C4	ERROR ADDRESSING R2		1D2	ERROR IN CONTROL STORE PARITY ERROR
	0C7	ERROR ADDRESSING R4			
	0C8	ERROR ADDRESSING R8	'^'		CACHE TEST
	0CE	ERROR ADDRESSING DUAL PORT		1E1	ERROR FILLING CACHE WITH ONES. LOCATION NOT INITIALLY = 0
				1E2	ERROR FILLING CACHE WITH ONES. UNABLE TO WRITE ONES

```
;1312    .TOC "  Micro Level Charts              : Micro Code Label Prefixes"
;1313
;1314    ;   Label ID    Micro Code Description
;1315
;1316    ;       BO      Bootstrap
;1317
;1318    ;       CH      Change Mode
;1319
;1320    ;       CM      Compatability Mode
;1321
;1322    ;       CN      Console Command Parser
;1323
;1324    ;       CO      Control
;1325
;1326    ;       CR      Cyclic Redundancy Check
;1327
;1328    ;       CS      Character String
;1329
;1330    ;       DS      Decimal String
;1331
;1332    ;       ED      Edit
;1333
;1334    ;       FI      FPA Interface
;1335
;1336    ;       FP      Floating Point
;1337
;1338    ;       FX      Floating Point Extension For G and H Data Types
;1339
;1340    ;       GL      Global Code
;1341
;1342    ;       IE      Interrupts and Exceptions
;1343
;1344    ;       II      Integer, Logical, & Address
;1345
;1346    ;       IN      Initialize
;1347
;1348    ;       LS      Load and Save Processor Context
;1349
;1350    ;       MM      Memory Management
;1351
;1352    ;       MP      Move To and From Processor Register
;1353
;1354    ;       MS      Miscellaneous
;1355
;1356    ;       MV      Micro Verity
;1357
;1358    ;       OS      Native Mode Operand Specifier
;1359
;1360    ;       PC      Procedure Call
;1361
;1362    ;       PR      Probe
;1363
;1364    ;       QU      Queue
;1365
;1366    ;       VF      Variable Length Bit Field
```

```
1367    .TOC "  Micro Level Charts              : Fixed Control Store Address"
1368
1369
1370    ;     +------+-----------------------------------+-----------------------------+
1371    ;     | Add  | Function of Vector                | Method of Initiation        |
1372    ;     +------+-----------------------------------+-----------------------------+
1373    ;     | 0000 | Power Up                          |                             |
1374    ;     | 0011 | Arithmetic Trap                   | DO Service                  |
1375    ;     | 0012 | FPA Integer Overflow Trap         | DO Service                  |
1376    ;     | 0014 | Timer Service                     | DO Service                  |
1377    ;     | 0015 | T-Bit trap                        | DO Service                  |
1378    ;     | 0016 | Console ^P Trap                   | DO Service                  |
1379    ;     |      |                                   |                             |
1380    ;     | 0020 | Control Store Parity Error        | Micro Trap                  |
1381    ;     | 0021 | Read Unaligned Data               | Micro Trap                  |
1382    ;     | 0022 | MSRC XB Miss                      | Micro Trap                  |
1383    ;     | 0023 | MSRC XB ACV                       | Micro Trap                  |
1384    ;     | 0024 | Write Unlock Unaligned Data       | Micro Trap                  |
1385    ;     | 0025 | Write Unaligned Data              | Micro Trap                  |
1386    ;     | 0026 | Write Unlock Crossing Page Boundry| Micro Trap                  |
1387    ;     | 0027 | Write Crossing Page Boundry       | Micro Trap                  |
1388    ;     | 0028 | Machine Check Exceptions(See Note)| Micro Trap                  |
1389    ;     | 0029 | But XB Miss                       | Micro Trap                  |
1390    ;     | 002A | Read TB Miss                      | Micro Trap                  |
1391    ;     | 002B | Write TB Miss                     | Micro Trap                  |
1392    ;     | 002C | FPA Reserved Operand              | Micro Trap                  |
1393    ;     | 002D | But XB ACV                        | Micro Trap                  |
1394    ;     | 002E | Read ACV                          | Micro Trap                  |
1395    ;     | 002F | Write ACV                         | Micro Trap                  |
1396    ;     |      |                                   |                             |
1397    ;     | 0038 | Soft Interrupt                    | DO Service, Execution Flows |
1398    ;     | 0039 | Console Interrupt                 | DO Service, Execution Flows |
1399    ;     | 003A | Unibus  Interrupt                 | DO Service, Execution Flows |
1400    ;     | 003B | Interval Timer Interrupt          | DO Service, Execution Flows |
1401    ;     | 003C | Corrected Memory Interrupt        | DO Service, Execution Flows |
1402    ;     | 003E | Write Bus Error Interrupt         | DO Service, Execution Flows |
1403    ;     | 003F | Power Fail                        | DO Service, Execution Flows |
1404    ;     +------+-----------------------------------+-----------------------------+
1405    ;
1406    ;     NOTE :  MSRC XB TB Error
1407    ;             MSRC XB Bus Error
1408    ;             Bus Error
1409    ;             Unaligned UNIBUS Data
1410    ;             TB Error
1411    ;             But XB TB Error
1412    ;             But XB Bus Error
```

```
1413    .TOC "  Micro Level Charts              : BUT/UVCTR Chart"
1414
1415
1416    ; BUT/UVCTR is not valid During any micro cycle that follows an IRD1.
1417    ; Because of this, it cannot be used in any of the micro instructions that are pointed to by the IRD1 rom.
1418
1419    ;
1420    ;
1421    ;
1422    ;
1423    ;
1424    ;
1425    ;
1426    ;
1427    ;
1428    ;
1429    ;
1430    ;
1431    ;
1432    ;
1433    ;
1434    ;
1435    ;
1436    ;
1437    ;
1438    ;
1439    ;
1440    ;
1441    ;
1442    ;
1443    ;
1444    ;
1445    ;
1446    ;
1447    ;
1448    ;
1449    ;
1450    ;
1451    ;
1452    ;
1453    ;
1454    ;
1455    ;
1456    ;
1457    ;
1458    ;
1459    ;
1460    ;
1461    ;
1462    ;
1463    ;
```

	UVCTR<3>	UVCTR<2>	UVCTR<1>	UVCTR<0>
BUS/PRB.RD BUS/PRB.RD.MODE BUS/PRB.WR BUS/PRB.WR.MODE	PBOK.AND.V.AND.AC).OR.PA	M.AND.((V.AND.AC).OR.PA)	V.OR.PA	(AC.AND.V).OR.PA
BUS/PRB.RD.PTE BUS/PRB.RD.PTE.K BUS/PRB.WR.PTE	0	M.AND.V.AND.AC	V.AND.AC	AC
WCTRL/UVCTR_CM.IS	1	PSL<IS>	PSL<CUR>	
WCTRL/REICHK and Wbus = Saved Psl	1	0	0 = REI Check Ok 1 = REI Check Ok & AST Pending 2 = REI Check is not Ok 3 = REI Check is not Ok	
WCTRL is not UVCTR_CM.IS or REICHK(see above) BUS is not one the Probe Micro Orders(see above)	1	0 = Soft Interrupt 1 = Console Interrupt 2 = Unibus Interrupt 3 = Interval Timer Interrupt 4 = Corrected Memory Interrupt 6 = Write Bus Frror Interrupt 7 = Power Fail Interrupt		

```
1459    ; Legend : M    = PTE Modify Bit
1460    ;          V    = 1 if Valid PTE
1461    ;          AC   = 1 if Access Allowed
1462    ;          PBOK = 1 if Not Crossing a Page Boundry
1463    ;          PA   = 1 if Memory Mapping is not Enabled
```

```
;1464    .TOC "  Micro Level Charts                : Description of FPD "
;1465
;1466    ;  When PSL<FPD> is set, there are 4 things that might require the instruction
;1467    ;  flow to be interrupted and the instruction packed up.
;1468    ;
;1469    ;       1) Memory Fault (ACV or TNV),
;1470    ;       2) Reserved Operand Fault,
;1471    ;       3) Floating Point Fault (overflow, underflow), or
;1472    ;       4) Interrupt.
;1473    ;
;1474    ;  In the first case the memory management flows will branch to the
;1475    ;  pack routine at IE.FPD.PACK + FPDOFFSET, and initiate the fault.
;1476    ;
;1477    ;  In the second case the micro code will branch to IE.OPER.FAULT which
;1478    ;  will branch to the pack routine and initiate the fault.
;1479    ;
;1480    ;  In the third case the micro code will branch to IE.FLxxx.FAULT which
;1481    ;  will branch to the pack routine and initiate the fault.
;1482    ;
;1483    ;  In the fourth case when the micro code decides to allow an interrupt it will
;1484    ;  branch to IE.SERV.IP.IS which will branch to the pack routine and initiate
;1485    ;  the interrupt (interrupts may be taken by the memory management flows
;1486    ;  if a TB miss occurs and the flag MM.NOINT is clear).
;1487    ;
;1488    ;  The micro code pack routines loaded at IE.FPD.PACK + FPDOFFSET observe
;1489    ;  the following rules for packing:
;1490    ;
;1491    ;       1) When packing is complete the pack routine will branch
;1492    ;          to IE.PACK.DONE.
;1493    ;
;1494    ;       2) For faults, DREG is modified by IANDE before going to the
;1495    ;          pack routine and must not be modified by the pack routine.
;1496    ;
;1497    ;       3) For floating faults and reserved operand faults, FLAG2 is modified
;1498    ;          by IANDE before going to the pack routine and must not be modified
;1499    ;          by the pack routine.
;1500    ;
;1501    ;       4) Micro temp FPDOFFSET<16:8> is modified by IANDE before going to the
;1502    ;          pack routine and must not be modified by the pack routine
;1503    ;          (FPDOFFSET<7:0> will be preserved).
;1504    ;
;1505    ;       5) In some cases the pack routine may need to return to the instruction
;1506    ;          flows before branching to IE.PACK.DONE, if so the address at the
;1507    ;          top of the micro stack will have one of two possible values:
;1508    ;               a) IF the pack routine was branched to because of a memory
;1509    ;                  management fault or because of an interrupt taken from a
;1510    ;                  tbmiss routine (MM.NOINT clear)
;1511    ;                  THEN the address at the top of the micro stack is the
;1512    ;                  address of the micro trapped instruction that caused the
;1513    ;                  tb miss.
;1514    ;               b) IF the pack routine was reached by any other path
;1515    ;                  (ie. NEXT/IE.SERV.IP.IS or NEXT/IE.OPER.FAULT)
;1516    ;                  THEN the address at the top of the micro stack is the address
;1517    ;                  of the last unmatched PUSH in the instruction flows.
```

```
;1518    .TOC "  Micro Level Charts                  : Compatability Mode Condition Codes"
;1519
;1520    ; CCOPS are not valid During any micro cycle that follows an IRD1. Because of this, they cannot be used in any of the micro
;1521    ; instructions that are pointed to by the IRD1 entries in the Compatibility mode rom.
;1522
;1523    ; NOTE 1 : 'SIGN', 'WX', 'OV', 'CRY', ARE ALL FUNCTIONS OF DSIZE
;1524    ; NOTE 2 : (SIGN.XOR.OV).OR.CRY
;1525
;1526    ;
```

INSTRUCTION	N	Z	V	C	CCOPx
ADC(B)	SIGN	WX.EQ.0	OV	CRY	1
ADD	SIGN	WX.EQ.0	OV	CRY	1
ASH	SIGN	WX.EQ.0	0	0	1
ASHC	SIGN	WX.EQ.0	0	0	1
	SIGN	(WX.EQ.0).AND.Z	0	C	2
ASL	SIGN	WX.EQ.0	N.XOR.C(IN)	WB<31:16>.NE.0	2
ASLB	SIGN	WX.EQ.0	N.XOR.C(IN)	WB<31:8>.NE.0	2
ASR(B)	SIGN	WX.EQ.0	N.XOR.C(IN)	WB<31>	1
BI(T,S,C)(B)	SIGN	WX.EQ.0	0	C	1
CLR(B)	SIGN	WX.EQ.0	0	0	1
CMP(B)	SIGN	WX.EQ.0	OV	.NOT.CRY	2
COM(B)	SIGN	WX.EQ.0	OV	.NOT.CRY	2
DEC(B)	SIGN	WX.EQ.0	OV	C	1
DIV	SIGN	WX.EQ.0	OV	CRY	1
INC(B)	SIGN	WX.EQ.0	OV	C	1
MFP(I,D)	SIGN	WX.EQ.0	0	C	1
MTP(I,D)	SIGN	WX.EQ.0	0	C	1
MOV(B)	SIGN	WX.EQ.0	0	C	1
MUL	SIGN	WX.EQ.0	0	0	1
	NOTE 2	(WX.EQ.0).AND.Z	0	WB<L>.NE.0	2
NEG(B)	SIGN	WX.EQ.0	OV	.NOT.CRY	2
ROL	SIGN	WX.EQ.0	N.XOR.C(IN)	WB<31:16>.NE.0	2
ROLB	SIGN	WX.EQ.0	N.XOR.C(IN)	WB<31:8>.NE.0	2
ROR(B)	SIGN	WX.EQ.0	N.XOR.C(IN)	WB<31>	1
SBC(B)	SIGN	WX.EQ.0	OV	.NOT.CRY	2
SUB	SIGN	WX.EQ.0	OV	.NOT.CRY	2
SWAB	SIGN	WX.EQ.0	0	0	1
SXT	SIGN	WX.EQ.0	0	C	1
TST(B)	SIGN	WX.EQ.0	0	0	1
XOR	SIGN	WX.EQ.0	0	C	1

```
;1559    ;
```

```
;1560    .TOC "  Micro Level Charts                : Native Mode Condition Codes Part 1"
;1561
;1562
;1563    ; CCOPS are not valid During any micro cycle that follows an IRD1. Because of this, they cannot be used in any of the micro
;1564    ; instructions that are pointed to by the IRD1 rom.
;1565    ; NOTE 1 : 'SIGN', 'WX', 'OV', 'CRY', ARE ALL FUNCTIONS OF DSIZE
;1566    ; NOTE 2 : WB<15> + [(WX<15:0>.EQ.0).AND.CRY]
;1567
;1568    ;  +-------------------+-------------+-------------- -+-----------+----------+-------+
;1569    ;  | INSTRUCTION       |      N      |       Z        |     V     |    C     | CCOPx |
;1570    ;  +-------------------+-------------+----------------+-----------+----------+-------+
;1571    ;  | ACB(B,W)          | SIGN        | WX.EQ.0        | OV        | C        |   1   |
;1572    ;  | ACBL              | SIGN        | WX.EQ.0        | OV        | C        |   2   |
;1573    ;  | ACB(F,D)          | WB<15>      | WX.EQ.0        | 0         | C        |   2   |
;1574    ;  | ADAWI             | SIGN        | WX.EQ.0        | OV        | CRY      |   2   |
;1575    ;  | ADD(B,W,L)(2,3)   | SIGN        | WX.EQ.0        | OV        | CRY      |   1   |
;1576    ;  | ADD(F,D)(2,3)     | WB<15>      | WX.EQ.0        | 0         | 0        |   1   |
;1577    ;  | ADWC              | SIGN        | WX.EQ.0        | OV        | CRY      |   1   |
;1578    ;  | AOB(LEQ,LSS)      | SIGN        | WX.EQ.0        | OV        | C        |   2   |
;1579    ;  | ASHL              | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1580    ;  | ASHQ              | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1581    ;  |                   | SIGN        | (WX.EQ.0).AND.Z | 0        | C        |   2   |
;1582    ;  | BIC(B,W,L)(2,3)   | SIGN        | WX.EQ.0        | 0         | C        |   2   |
;1583    ;  | BIS(B,W,L)(2,3)   | SIGN        | WX.EQ.0        | 0         | C        |   2   |
;1584    ;  | BIT(B,W,L)        | SIGN        | WX.EQ.0        | 0         | C        |   2   |
;1585    ;  | CASE(B,W,L)       | SIGN.XOR.OV | WX.EQ.0        | 0         | .NOT.CRY |   1   |
;1586    ;  | CLR(B,W,L)        | SIGN        | WX.EQ.0        | 0         | C        |   2   |
;1587    ;  | CLRD              | SIGN        | WX.EQ.0        | 0         | C        |   1   |
;1588    ;  | CLRF              | SIGN        | WX.EQ.0        | 0         | C        |   2   |
;1589    ;  | CLRQ              | SIGN        | WX.EQ.0        | OV        | C        |   1   |
;1590    ;  | CMP(B,W,L)        | SIGN.XOR.OV | WX.EQ.0        | 0         | .NOT.CRY |   1   |
;1591    ;  | CMP(V,ZV)         | SIGN.XOR.OV | WX.EQ.0        | 0         | .NOT.CRY |   1   |
;1592    ;  | CMPC(3,5)         | SIGN.XOR.OV | WX.EQ.0        | 0         | .NOT.CRY |   1   |
;1593    ;  | CMPD              | SEE NOTE 2  | WX.EQ.0        | 0         | 0        |   2   |
;1594    ;  |                   | .NOT.CRY    | WX.EQ.0        | 0         | 0        |   1   |
;1595    ;  | CMPF              | WB<15>      | WX.EQ.0        | 0         | 0        |   1   |
;1596    ;  |                   | SEE NOTE 2  | WX.EQ.0        | 0         | 0        |   2   |
;1597    ;  | CRC               | SIGN        | WX.EQ.0        | 0         | C        |   2   |
;1598    ;  | CVT(BW,BL)        | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1599    ;  | CVT(FB,DB,FW,DW)  | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1600    ;  |                   | N           | Z              | WX<L>.NE.0 | C        |   2   |
;1601    ;  | CVT(FD,DF,BF,BD,  | WB<15>      | WX.EQ.0        | 0         | 0        |   1   |
;1602    ;  |     WF,WD,LD,LF)  |             |                |           |          |       |
;1603    ;  | CVT(FL,DL,        | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1604    ;  |     RFL,RDL)      |             |                |           |          |       |
;1605    ;  | CVT(LP,PL)        | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1606    ;  | CVT(WB,LB,LW)     | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1607    ;  |                   | N           | Z              | WX<L>.NE.0 | C        |   2   |
;1608    ;  | CVTWL             | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1609    ;  | DEC(B,W,L)        | SIGN        | WX.EQ.0        | OV        | .NOT.CRY |   1   |
;1610    ;  | DIV(B,W,L)(2,3)   | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1611    ;  | DIV(F,D)(2,3)     | WB<15>      | WX.EQ.0        | 0         | 0        |   1   |
;1612    ;  | EDIV              | SIGN        | WX.EQ.0        | 0         | 0        |   1   |
;1613    ;  | FMOD(F,D)         | WB<15>      | WX.EQ.0        | 0         | 0        |   1   |
;1614    ;  +-------------------+-------------+----------------+-----------+----------+-------+
```

```
:1615   .TOC "  Micro Level Charts                : Native Mode Condition Codes Part 2"
:1616
:1617   ; 'SIGN', 'WX', 'OV', 'CRY', ARE ALL FUNCTIONS OF DSIZE
:1618
:1619   ; +-----------------+-----------+---------------+----------+----------+-------+
:1620   ; | INSTRUCTION     |     N     |       Z       |     V    |     C    | CCOPx |
:1621   ; +-----------------+-----------+---------------+----------+----------+-------+
:1622   ; | EMUL            | SIGN      | WX.EQ.0       | 0        | 0        |   1   |
:1623   ; |                 | SIGN      | (WX.EQ.0).AND.Z | 0      | C        |   2   |
:1624   ; | EXT(V,ZV)       | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1625   ; | FF(S,C)         | 0         | WX.EQ.0       | 0        | 0        |   1   |
:1626   ; | INC(B,W,L)      | SIGN      | WX.EQ.0       | OV       | CRY      |   1   |
:1627   ; | INDEX           | SIGN      | WX.EQ.0       | 0        | 0        |   2   |
:1628   ; | INSQUE          | SIGN.XOR.OV | WX.EQ.0     | 0        | .NOT.CRY |   1   |
:1629   ; | INSV            | SIGN      | WX.EQ.0       | OV       | C        |   2   |
:1630   ; | LOCC            | 0         | WX.EQ.0       | 0        | 0        |   1   |
:1631   ; | M(T,F)PR        | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1632   ; | MATCH           | 0         | WX.EQ.0       | 0        | 0        |   1   |
:1633   ; | MCOM(B,W,L)     | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1634   ; | MNEG(B,W,L)     | SIGN      | WX.EQ.0       | OV       | .NOT.CRY |   1   |
:1635   ; | MNEG(F,D)       | WB<15>    | WX.EQ.0       | 0        | 0        |   1   |
:1636   ; | MOV(B,W,L)      | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1637   ; | MOV(F,D)        | WB<15>    | WX.EQ.0       | 0        | C        |   2   |
:1638   ; | MOVA(B,W,L)     | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1639   ; | MOVAQ           | SIGN      | WX.EQ.0       | 0        | C        |   1   |
:1640   ; | MOVC(3,5)       | 0         | WX.EQ.0       | 0        | 0        |   2   |
:1641   ; |                 | SIGN.XOR.OV | WX.EQ.0     | 0        | .NOT.CRY |   1   |
:1642   ; | MOVQ            | SIGN      | WX.EQ.0       | OV       | C        |   1   |
:1643   ; |                 | SIGN      | (WX.EQ.0).AND.Z | 0      | C        |   2   |
:1644   ; | MOVTC           | SIGN.XOR.OV | WX.EQ.0     | 0        | .NOT.CRY |   1   |
:1645   ; | MOVTUC          | SIGN.XOR.OV | WX.EQ.0     | 0        | .NOT.CRY |   1   |
:1646   ; | MOVZ(BW,BL)     | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1647   ; | MOVZWL          | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1648   ; | MUL(B,W,L)(2,3) | SIGN      | WX.EQ.0       | 0        | 0        |   1   |
:1649   ; |                 | N         | Z             | WX<L>.NE.0 | C      |   2   |
:1650   ; | MUL(F,D)(2,3)   | WB<15>    | WX.EQ.0       | 0        | 0        |   1   |
:1651   ; | POLY(F,D)       | WB<15>    | WX.EQ.0       | 0        | 0        |   1   |
:1652   ; | PROBE(R,W)      | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1653   ; | PUSHA(B,W,L)    | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1654   ; | PUSHAQ          | SIGN      | WX.EQ.0       | 0        | C        |   1   |
:1655   ; | PUSHL           | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1656   ; | REMQUE          | SIGN.XOR.OV | WX.EQ.0     | 0        | .NOT.CRY |   1   |
:1657   ; |                 | N         | Z             | WX<L>.EQ.0 | C      |   2   |
:1658   ; | ROTL            | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1659   ; | S(C,P)ANC       | 0         | WX.EQ.0       | 0        | 0        |   2   |
:1660   ; | SBWC            | SIGN      | WX.EQ.0       | OV       | .NOT.CRY |   1   |
:1661   ; | SKPC            | 0         | WX.EQ.0       | 0        | 0        |   1   |
:1662   ; | SOB(GEQ,GTR)    | SIGN      | WX.EQ.0       | OV       | C        |   2   |
:1663   ; | SUB(B,W,L)(2,3) | SIGN      | WX.EQ.0       | OV       | .NOT.CRY |   2   |
:1664   ; | SUB(F,D)(2,3)   | WB<15>    | WX.EQ.0       | 0        | 0        |   1   |
:1665   ; | TST(B,W,L)      | SIGN      | WX.EQ.0       | 0        | 0        |   1   |
:1666   ; | TST(F,D)        | WB<15>    | WX.EQ.0       | 0        | 0        |   1   |
:1667   ; | XOR(B,W,L)(2,3) | SIGN      | WX.EQ.0       | 0        | C        |   2   |
:1668   ; +-----------------+-----------+---------------+----------+----------+-------+
```

```
.TOC "  Micro Level Charts                : Native Mode Operand Specifier Chart"
```

	OS.RED	OS.MOD	OS.ADD	OS.VADD	OS.WRT1	OS.WRT2	OS.FRED	OS.QRED	OS.DMOD	OS.DRED
Rn	1	1	ERROR	1	1	1	1	2	2	2
(Rn)	2	2	1	1	1	1	2	3	4	3
(Rn)+	2	2	2	2	2	2	2	3	4	3
(PC)+	2	UNP	4	2	UNP	UNP	2	2	UNP	2
@(Rn)+	4	4	4	2	3	3	4	5	6	5
@(PC)+	2	2	2	1	1	2	2	3	4	3
-(Rn)	2	2	2	1	1	2	2	3	4	3
X^D(Rn)	2	2	2	1	1	2	2	3	4	3
X^D(PC)	2	2	3	1	1	2	2	3	4	3
@X^D(Rn)	4	4	2	2	3	3	4	5	6	5
@X^D(PC)	4	4	2	2	3	3	4	5	6	5
S^#Lit	1	ERROR	ERROR	ERROR	ERROR	ERROR	1	2	ERROR	3
INDEX	4	4	4	4	4	4	4	5	6	5
PC INDEX	ERROR	ERROR	ERROR	ERROR	ERROR	ERROR	ERROR	ERROR	ERROR	ERROR
MDR	OPERAND	OPERAND	OPERAND	OPER 1	UNP 3	UNP 3	OPERAND	MSB OPER	MSB OPER	MSB OPER
Q-REG	MDR OLD	MDR OLD	MDR OLD			MDR OLD	MDR OLD	MDR OLD		
TEMP0	UNP	UNP	UNP	UNP	UNP	UNP	UNP	UNP	UNP	UNP
TEMP1								LSB OPER	LSB OPER	LSB OPER
VA/RNUM	UNP 3	ADD 3	UNP	UNP 2	ADD 3	ADD 3	UNP 3	UNP 3	ADD 3	UNP 3
	RET+1	RET+1	RET+1	RET+3	RET+1	RET+1	RET+1	RET+1	RET+1	RET+1

UNP : UNPREDICTABLE
ADD : Address of the Operand is left in VA if the Operand is in memory and in RNUM if the Operand is in GPR
NOTE :
1. In reg mode MDR <- GPR(RNUM), 2. In reg mode RNUM <- # of addressed reg, 3. In reg mode MDR & VA are not affected

```
1724    .TOC "  Micro Level Charts              :       Native Mode Addressing Branch"
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
```

Branch Offset	Operand Specifier Mode	Operand Specifier Reg	Addressing Mode	
0000	5	0-F	Rn	REGISTER MODE
0001	8	0-E	(Rn)+	AUTOINCREMENT MODE
0010	8	F	I^#cons	IMMEDIATE MODE
0011	0-3	---	S^#cons	LITERAL MODE
0100	7	0-F	-(Rn)	AUTODECREMENT MODE
0101	A,C,E	F	Addr	RELATIVE MODE
0110	A,C,E	0-E	D(Rn)	DISPLACEMENT MODE
0111	9	F	@#Addr	ABSOLUTE MODE
1000	6	0-F	(Rn)	REGISTER DEFERRED MODE
1001	B,D,F	F	@Addr	RELATIVE DEFERRED MODE
1010	B,D,F	0-E	@D(Rn)	DISPLACEMENT DEFERRED MODE
1011	9	0-E	@(Rn)+	AUTO-INCREMENT DEFERRED MODE
1100	4	F	(Rn)[PC]	INDEX MODE PC
1101	4	0-E	(Rn)[Rx]	INDEX MODE

```
1764    .TOC "  Micro Level Charts                :       Compatibility Mode Addressing Branch"
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
```

Branch Offset	Operand Specifier Mode	Operand Specifier Reg	Addressing Mode	
0000	0	0-6	Rn	REGISTER MODE
0001	0	7	PC	REGISTER MODE PC
0010	1	0-6	(Rn)	REGISTER DEFERRED MODE
0011	1	7	(PC)	REGISTER DEFERRED MODE PC
0100	2	0-5	(Rn)+	AUTOINCREMENT MODE
0101	2	6	(SP)+	AUTOINCREMENT MODE SP
0110	3	0-6	@(Rn)+	AUTO-INC DEFERRED MODE
0111	3	7	@#Addr	ABSOLUTE MODE
1000	4	0-5	-(Rn)	AUTO-DECREMENT MODE
1001	4	6	-(SP)	AUTO-DECREMENT MODE SP
1010	5	0-7	@-(Rn)	AUTO-DEC DEFERRED MODE
1011	4	7	-(PC)	AUTO-DECREMENT MODE PC
1100	6	0-6	X(Rn)	INDEX MODE
1101	6	7	Addr X(PC)	RELATIVE MODE
1110	7	0-7	@ADDR @X(Rn)	INDEX DEFERRED MODE
1111	2	7	#CONS	IMMEDIATE MODE

```
1807    .TOC "  Micro Level Charts              : WBUS Drive Chart"
1808
1809    ;
1810    ;
1811    ;
1812    ;
1813    ;
1814    ;
1815    ;
1816    ;
1817    ;
1818    ;
1819    ;
1820    ;
1821    ;
1822    ;
1823    ;
1824    ;
1825    ;
1826    ;
1827    ;
1828    ;
1829    ;
1830    ;
1831    ;
1832    ;
1833    ;
1834    ;
1835    ;
1836    ;
1837    ;
1838    ;
1839    ;
1840    ;
1841    ;
1842    ;
1843    ;
1844    ;
1845    ;
1846    ;
1847    ;
1848    ;
1849    ;
1850    ;
1851    ;
1852    ;
1853    ;
```

FIELDS THAT DRIVE WBUS	
ALPCTL ALU ALUOD DQ1 DQ2 DQ3 LIT MUX	ALU GROUP
WCTRL CCMISC CCPSL	WCTRL GROUP
FPA MSRC	OTHERS

ALU Group Output Disable Chart
(Always enabled when LONLIT)

ALU	MUX=09	MUX=0D
08	Disable	
09	Disable	Disable
0A	Disable IF DQ=0	Disable
0B	Disable IF DQ=1	Disable
0C	Disable	
0D	Disable	Disable
0E	Disable IF DQ=0	
0F	Disable IF DQ=1	

WCTRL Group Drive Micro Orders

WCTRL	CCMISC	CCPSL	
03			FPTCR
		04	WB_PSL.CCBR_SIGND
	07		WB_ATCR.CCBR_SIGND
0C			CM.TP.FPD.F5.STEPC
0D			TODCLK
0E			INIR
0F			TCSR.IICR
11			REVLEVEL
1A			READCRAR
1B			READTRAR
1C			CM.TP.FPD.FLAGS
1D			ACLOSYNC
1E			CONREAD
1F			TU58READ
32			MEMSCR
3A			ASTLVL
3F			IPR

Others Drive Micro Orders

FPA	MSRC	
04		WBUS_FPA
05		WBUS_FPA.CC
	16	WBUS_RNUM
	1C	READRBS
	1E	WB_RBSP

```
;1854    .TOC "  Micro Level Charts                  : WCTRL/CCPSL vs BUS Micro Order Conflict Chart"
;1855
;1856    ; V000 : (04) WB_PSL.CCBR_SIGND
;1857    ;        (21) RESERVED
;1858    ;        (22) VA_VA+4
;1859    ;        (24) PC_WB
;1860    ;        (25) VA_WB
;1861    ;        (2C) PC_PC+WB
;1862    ;        (31) PREV_WB
;1863    ;        (34) MEMSCAR_WB
;1864    ;        (38) ASTLVL_WB
;1865    ;        (3A) ASTLVL
;1866    ;        (3C) SOFTIPR_WB
;1867    ;        (3D) IPL_WB
;1868    ;        (3F) IPR
;1869    ;
;1870    ; V001 : (00) PSL_WB.CCBR_ALUS
;1871    ;        (20) VA_PC+I+W.PC_PC+I
;1872    ;        (30) MEMSCR_WB
;1873    ;        (32) MEMSCR
;1874    ;        (35) PREV_CUR.ISCUR_WB
;1875    ;
;1876    ; V002 : (23) MDR_WB
;1877    ;        (27) MDR_0
;1878    ;        (2A) WDR_WB.UR
;1879    ;        (2F) MDR_OSR.CCBR_BRATST
;1880    ;        (2E) WDR_WB
;1881    ;        (28) MDR_IR
;1882    ;
;1883    ;
;1884    ; V004 : (20) CLRCH.VA_WB
;1885    ;
;1886    ; V005 : (37) REICHK
;1887    ;
;1888    ; V006 : (33) GRANT
;1889    ;
;1890    ; V007 : (26) MBUS_WDR
;1891    ;
;1892    ; V010 : (29) CLRTB.VA_WB
;1893    ;
;1894    ; V011 : (28) TB_WB
;1895    ;
;1896    ;
;1897
;1898    ; ***  VALID IF BUT.NE.UVCTR
```

	V 000	V 001	V 002	V 004	V 005	V 006	V 007	V 010	V 011
(00) READ.PHY		X	X	X	X	X	X	X	X
(01) PRINIT	X	X	X	X	X	X	X	X	X
(02) READ.NT		X	X	X	X	X	X	X	X
(03) IOINIT	X	X	X	X	X	X	X	X	X
(04) RESERVED		X	X	X	X	X	X	X	X
(05) RESERVED					X	X			
(06) READ.SEC		X	X	X	X	X	X	X	X
(07) NOP					X	X			
(08) WRITE.PHY		X		X	X	X	X	X	X
(09) REICHK	X	X	X	X		X	X	X	X
(0A) WRITE.SEC		X		X	X	X	X	X	X
(0B) WRITE.UL.SEC		X		X	X	X	X	X	X
(0C) WRITE.NT		X		X	X	X	X	X	X
(0D) RESERVED					X	X			
(0E) WRITE.NT.LNG		X		X	X	X	X	X	X
(0F) GRANT	X	X	X	X	X		X	X	X
(10) READ		X	X	X	X	X	X	X	X
(11) READ.LNG		X	X	X	X	X	X	X	X
(12) PRB.WR.PTE			X	X	X	X	X	X	X
(13) READ.MOD.LCK		X	X	X	X	X	X	X	X
(14) READ.MOD		X	X	X	X	X	X	X	X
(15) READ.LNG.MOD		X	X	X	X	X	X	X	X
(16) PRB.RD.PTE			X	X	X	X	X	***	X
(17) PRB.RD.PTE.K			X	X	X	X	X	X	X
(18) WRITE		X		X	X	X	X	X	X
(19) WRITE.LNG		X		X	X	X	X	X	X
(1A) WRITE.NOREG		X		X	X	X	X	X	X
(1B) WRITE.UL		X		X	X	X	X	X	X
(1C) PRB.WR.MODE				X	X	X		X	X
(1D) PRB.WR				X	X	X		X	X
(1E) PRB.RD.MODE				X	X	X		X	X
(1F) PRB.RD				X	X	X		X	***

```
;1899    .TOC "  Micro Level Charts              : BUS vs MSRC Micro Order Conflict Chart"
;1900
;1901    ; V020 : (08) WRITE.PHY
;1902    ;        (0A) WRITE.SEC
;1903    ;        (0B) WRITE.UL.SEC
;1904    ;        (0C) WRITE.NT
;1905    ;        (0E) WRITE.NT.LNG
;1906    ;        (18) WRITE
;1907    ;        (19) WRITE.LNG
;1908    ;        (1A) WRITE.NOREG
;1909    ;        (1B) WRITE.UL
;1910    ;
;1911    ; V021 : (01) PRINIT
;1912    ;        (03) IOINIT
;1913    ;        (0F) GRANT
;1914    ;
;1915    ; V022 : (00) READ.PHY
;1916    ;        (02) READ.NT
;1917    ;        (06) READ.SEC
;1918    ;        (10) READ
;1919    ;        (11) READ.LNG
;1920    ;        (13) READ.MOD.LCK
;1921    ;        (14) READ.MOD
;1922    ;        (15) READ.LNG.MOD
;1923    ;
;1924    ; V023 : (12) PRB.WR.PTE
;1925    ;        (16) PRB.RD.PTE
;1926    ;        (17) PRB.RD.PTE.K
;1927    ;
;1928    ; V024 : (1C) PRB.WR.MODE
;1929    ;        (1D) PRB.WR
;1930    ;        (1E) PRB.RD.MODE
;1931    ;        (1F) PRB.RD
;1932    ;
;1933    ; V025 : (1C) PRB.WR.MODE
;1934    ;        (1D) PRB.WR
;1935    ;        (1E) PRB.RD.MODE
;1936    ;        (1F) PRB.RD
;1937    ;
;1938    ;
;1939    ;
;1940    ;
;1941    ;
;1942    ;
;1943    ; *** Valid if BUT.ne.UVCTR
;1944    ; @@@ Valid if in INIT code because MM is not set
```

	V020	V021	V022	V023	V024	V025
(00) TEMP0						
(01) TEMP1						
(02) TEMP2						
(03) TEMP3						
(04) TEMP4				X		X
(05) TEMP5				X		X
(06) TEMP6				X		X
(07) TEMP7				X		X
(08) TEMP8						
(09) TEMP9						
(0A) TEMP10						
(0B) TEMP11						
(0C) FPDOFFSET				X		X
(0D) MM.TEMP0				X		X
(0E) SCBB				X		X
(0F) SISR				X		X
(10) TEMP.R						
(11) TEMP.R+1						
(12) MDR		X				
(13) WDR		X	X	X		
(14) PSHSUB				X		X
(15) PSHADD				X		X
(16) WBUS_RNUM				X		X
(17) XB.PC_PC+1	@@@	X	@@@	X		X
(18) MA	X	X	X	X	***	
(19) PCBACK	X	X	X	X	***	
(1A) PC	X	X	X	X	***	
(1B) VA	X	X	X	X	***	
(1C) READRBS				X		X
(1D) RNUM_WBUS				X		X
(1E) WB_RBSP				X		X
(1F) TB	X	X	X	X	***	

```
.TOC "  Micro Level Charts                : WCTRL/CCPSL vs MSRC Micro Order Conflict Chart"

 V030 : (00) PSL_WB.CCBR_ALUS
        (04) WB_PSL.CCBR_SIGND
        (22) VA_VA+4
        (24) PC_WB
        (25) VA_WB
        (2C) PC_PC+WB
        (31) PREV_WB
        (34) MEMSCAR_WB
        (35) PREV_CUR.ISCUR_WB
        (37) REICAK
        (38) ASTLVL_WB
        (3A) ASTLVL
        (3C) SOFTIPR_WB
        (3D) IPL_WB
        (3F) IPR

 V031 : (20) VA_PC+I+W.PC_PC+I

 V032 : (21) RESERVED
        (30) MEMSCR_WB
        (32) MEMSCR

 V033 : (23) MDR_WB
        (2A) WDR_WB.UR
        (2F) MDR_OSR.CCBR_BRATST
        (2D) CLRCH.VA_WB
        (2E) WDR_WB
        (2B) MDR_IR

 V034 : (28) TB_WB

 V035 : (29) CLRTB.VA_WB

 V036 : (33) GRANT

 V037 : (26) MBUS_WDR

 V021 : (27) MDR_0
```

	V030	V031	V032	V033	V034	V035	V036	V037	V021
(00) TEMP0		X			X			X	
(01) TEMP1		X			X			X	
(02) TEMP2		X			X			X	
(03) TEMP3		X			X			X	
(04) TEMP4		X			X			X	
(05) TEMP5		X			X			X	
(06) TEMP6		X			X			X	
(07) TEMP7		X			X			X	
(08) TEMP8		X			X			X	
(09) TEMP9		X			X			X	
(0A) TEMP10		X			X			X	
(0B) TEMP11		X			X			X	
(0C) FPDOFFSET		X			X			X	
(0D) MM.TEMP0		X			X			X	
(0E) SCBB		X			X			X	
(0F) SISR		X			X			X	
(10) TEMP.R		X			X			X	
(11) TEMP.R+1		X			X			X	
(12) MDR		X			X		X	X	X
(13) WDR	X	X	X	X	X	X	X		X
(14) PSHSUB		X			X			X	
(15) PSHADD		X			X			X	
(16) WBUS_RNUM		X			X			X	
(17) XB.PC_PC+I			X		X		X	X	X
(18) MA		X			X			X	X
(19) PCBACK		X			X	X		X	X
(1A) PC		X			X	X		X	X
(1B) VA		X						X	X
(1C) READRBS		X			X			X	
(1D) RNUM_WBUS		X			X			X	
(1E) WB_RBSP		X			X			X	
(1F) TB		X		X	X		X	X	X

```
;1988   .TOC "  Micro Level Charts              : BUT/CCBR Chart"
;1989
;1990
;1991   ;
;1992   ;
;1993   ;
;1994   ;
;1995   ;
;1996   ;
;1997   ;
;1998   ;
;1999   ;
;2000   ;
;2001   ;
;2002   ;
;2003   ;
;2004   ;
;2005   ;
;2006   ;
;2007   ;
;2008   ;
;2009   ;
;2010   ;
;2011   ;
;2012   ;
;2013   ;
;2014   ;
;2015   ;
;2016   ;
;2017   ;
;2018   ;
;2019   ;
;2020   ;
;2021   ;
;2022   ;
;2023   ;
;2024   ;
;2025   ;
;2026   ;
;2027   ;
;2028   ;
;2029   ;
;2030   ;
;2031   ;
;2032   ;
;2033   ;
;2034   ;
```

	MICRO ORDER	OPERATION	CCBR CONTROL	
			CCBR<1>	CCBR<0>
	NOP.CCBR<-SIGND		LSS 0	EQL 0
C	NOP.CCBR<-ALUS		ALUS<1>	ALUS<0>
C	CCOP1.CCBR<-SIGND	CC OP 1	LSS 0	EQL 0
	CCOP2.CCBR<-SIGND	CC OP 2	LSS 0	EQL 0
C	NOP.CCBR<-BRATST		0	BRA TST
C	NOP.CCBR<-CSIGNS		ALUS<1>	LSS 0
M	WB<-ATCR.CCBR<-SIGND	WB<3:0> <-ATCR	LSS 0	EQL 0
I	ALUS<-DSDZ.CCBR<-ALUS	ALUS<1:0> <- (BCD SIGN)'(BCD 0)	ALUS<1>	ALUS<0>
S	ALUS<-SIGND.CCBR<-ALUS	ALUS<1:0> <- (LSS 0)'(EQL 0)	ALUS<1>	ALUS<0>
C	ALUS<-UNSGN.CCBR<-ALUS	ALUS<1:0> <- (LSSU 0)'(EQL 0)	ALUS<1>	ALUS<0>
	SETV.CCBR<-SIGND	PSL<V> <- 1	LSS 0	EQL 0
C	WB<-PSL.CCBR<-SIGND	WB<31:0> <-PSL	LSS 0	EQL 0
C	CC<-WB.CCBR<-ALUS	CC<-WB<3:0>	ALUS<1>	ALUS<0>
P	PSL<-WB.CCBR<-ALUS	PSL<-WB<31:0> ***	ALUS<1>	ALUS<0>
S	PSW<-WB.CCBR<-ALUS	PSW<-WB<15:0> ***	ALUS<1>	ALUS<0>
L	MDR_OSR.CCBR_BRATST	MDR <- ZEXT OSR	0	BRA TST

WHENEVER THE OPERATION ALTERS ALUS, CCBR <- ALUS OLD

UNLESS OTHERWISE NOTED ALL VALUES ARE SIZE DEPENDENT

LSS 0 : WBUS<SIGN>.XOR.ALU 'OVERFLOW' (THIS WILL PRODUCE 'LSS 0' FOR A-B OR A+(-B))

OVERFLOW : XOR OF CIN AND COUT OF MSB(SIZE) ALU

EQL 0 : WMUX = 0

LSSU 0 : .NOT.(ALU CARRY) (THIS WILL PRODUCE 'LSSU 0' FOR A-B OR A+(-B))

BCD 0 : (WMUX<31:8>.EQ.0).AND.(WBUS<7:4>.EQ.0)

BCD SIGN : IF WBUS<3:0> > 9 THEN, ALUS<1> IS SET IF SIGN IS NEGATIVE

BRA TST : FOR BRANCH INSTRUCTIONS CCBR<0> IS SET IF THE BRANCH CONDITION IS TRUE

*** IF V GETS SET NO TRAP WILL OCCUR

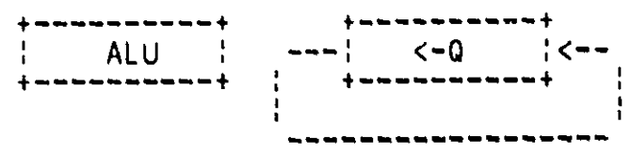
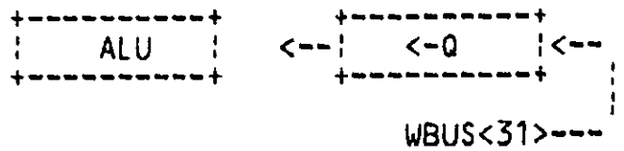
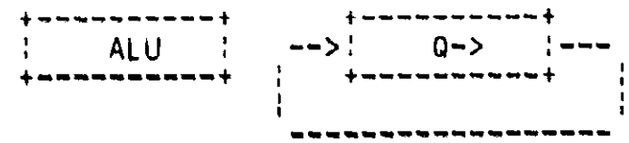
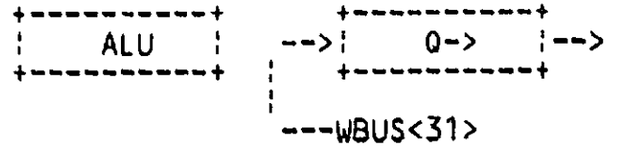
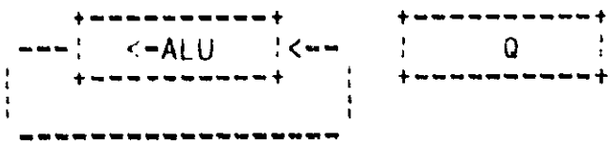
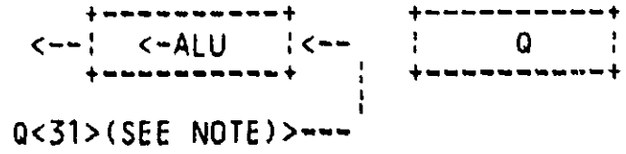
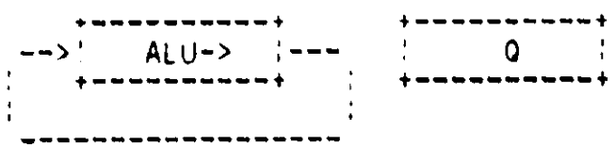
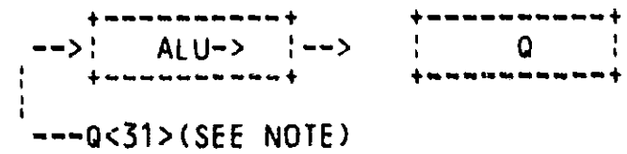
```
.TOC "  Micro Level Charts                : BUT/SPASTA Chart"
```

IPR DECODE & RBS STATUS			COMPATIBILITY REG DECODE & NATIVE REG=SP TEST		
RSRC does not Specify GPR[0,15]			RSRC Specifies GPR[0,15]		
RSRC : TEMP0	KSP	TEMP.R	RSRC : R0		GPR.R
TEMP1	ESP	TEMP.R+1	R1		GPR.R+1
TEMP2	SSP	TEMP.ROR1	R2		GPR.ROR1
TEMP3	USP	IPR.R	R3		DST.R
TEMP4	ISP	IPR.R+1	R4		DST.R+1
TEMP5	PCBB	IPR.ROR1	R5		DST.ROR1
TEMP6	MM.TEMP2	LONLIT	R6		
TEMP7	MM.TEMP3	ZERO	R7		
TEMP8	P0BR	ZERO.CLRRBSP	R8		
TEMP9	P0LR		R9		
TEMP10	P1BR		R10		
TEMP11	P1LR		R11		
TEMP12	SBR		R12		
TEMP13	SLR		R13		
MM.TEMP5	SPNICR.SPICR		SP		
MM.TEMP1	MM.TEMP4		RTMPGPR		

MSRC		SPASTA<1:0>	MSRC		SPASTA<1:0>
RNUM_WBUS	WBUS.EQ.[8,9,10,11,12,13]	00	RNUM_WBUS		UNDEFINED
RNUM_WBUS	WBUS.EQ.[5,6,7,14,15]	10	READRBS		UNDEFINED
RNUM_WBUS	WBUS.EQ.[0,1,2,3,4]	11	WB_RBSP		UNDEFINED
READRBS	RBS<6>.EQ.1(PSHADD)	10	ALL OTHERS	RNUM.NE.[6,7,14]	00
READRBS	RBS<6>.EQ.0(PSHSUB)	00	ALL OTHERS	RNUM.EQ.14	01
WB_RBSP	RBSP.EQ.0	01	ALL OTHERS	RNUM.EQ.7	10
WB_RBSP	RBSP.NE.0	00	ALL OTHERS	RNUM.EQ.6	11
ALL OTHERS		00			

```
;2085    .TOC "  Micro Level Charts              : TABLE 1 - ALU & Q Rotate and Shift Functions Part 1"
;2086
;2087
;2088    ; ALU  Q  |          ROTATE                          |          SHIFT
;2089    ;         |                                          |
;2090    ;         |    +----------+      +----------+       |    +----------+      +----------+
;2091    ;  L   L  | ---|  <-ALU   |<-----|   <-Q    |<--    | <--|  <-ALU   |<-----|   <-Q    |<--0
;2092    ;         |    +----------+      +----------+   |   |    +----------+      +----------+
;2093    ;         |                                     |   |
;2094    ;         | ------------------------------------    |
;2095    ;         |                                          |
;2096    ;         |                                          |
;2097    ;         |                                          |
;2098    ;         |                                          |
;2099    ;         |                                          |
;2100    ;         |                                          |
;2101    ;         |          +----------+                    |          +----------+
;2102    ;         |     ---  |  <-ALU   |<--                 |     <--  |  <-ALU   |<--
;2103    ;         |          +----------+   |                |          +----------+   |
;2104    ;  L   R  |                         |                |                         |
;2105    ;         |          +----------+   |                |          +----------+   |
;2106    ;         |     -->  |    Q->   |---                 |  0-->    |    Q->   |---
;2107    ;         |          +----------+                    |          +----------+
;2108    ;         |                                          |
;2109    ;         |                                          |
;2110    ;         |                                          |
;2111    ;         |                                          |
;2112    ;         |                                          |
;2113    ;         |                                          |
;2114    ;         |                                          |
;2115    ;         |          +----------+                    |          +----------+
;2116    ;         |     -->  |   ALU->  |---                 |  -->     |   ALU->  |-->
;2117    ;         |          +----------+   |                |          +----------+   |
;2118    ;  R   L  |                         |                |                         |
;2119    ;         |          +----------+   |                |          +----------+   |
;2120    ;         |     ---  |   <-Q    |<--                 |  ---     |   <-Q    |<--0
;2121    ;         |          +----------+                    |          +----------+
;2122    ;         |                                          |
;2123    ;         |                                          |
;2124    ;         |                                          |
;2125    ;         |                                          |
;2126    ;         |                                          |
;2127    ;         |                                          |
;2128    ;         |                                          |
;2129    ;         |    +----------+      +----------+       |    +----------+      +----------+
;2130    ;  R   R  | -->|  ALU->   |----->|    Q->   |---    | 0-->|  ALU->   |----->|    Q->   |-->
;2131    ;         |    +----------+      +----------+   |   |    +----------+      +----------+
;2132    ;         |                                     |   |
;2133    ;         | ------------------------------------    |
```

```
:2134   .TOC "  Micro Level Charts                : TABLE 2 - ALU & Q Rotate and Shift Functions Part 2"
```

ALU	Q	ROTATE	SHIFT
N	L		
N	R		
L	N		
R	N		

NOTE: Q<31> IS UNDEFINED FOR ANY LOAD Q FUNCTION

```
;2179    .TOC "  Micro Level Charts              : TABLE 2 - Console Interface Control"
;2180
;2181
;2182    ;       CRAR    : Console 'Register Address' Register
;2183    ;
;2184    ;       XBUF    : Xmitter data BUFfer
;2185    ;       RBUF    : Reciever data BUFfer
;2186    ;       XCSR    : Xmitter  Command and Status Register
;2187    ;       RCSR    : Reciever Command and Status Register
;2188    ;       CSR     : Command and Status Register
;2189    ;
;2190    ;
;2191    ;
;2192    ;
;2193    ;
;2194    ;
;2195    ;
;2196    ;
;2197    ;
;2198    ;
;2199    ;
;2200    ;
;2201    ;
;2202    ;
;2203    ;
;2204    ;
;2205    ;
;2206    ;
;2207    ;
;2208    ;
;2209    ;
;2210    ;
;2211    ;
;2212    ;
;2213    ;
;2214    ;
;2215    ;
;2216    ;
;2217    ;
```

	CRAR=0 (DATA)	CRAR=1 (XMIT STATUS)	CRAR=2 (RECV STATUS)	CRAR=3 (COMMAND & STATUS)
WCTRL/LOADCRAR	CRAR <- WBUS<23:22>			
WCTRL/CONWRITE	XBUF <- WBUS<23:16> (CLEARS READY)	XCSR<6> <- WBUS<22> INT ENABLE	RCSR<6> <- WBUS<22> INT ENABLE	CSR <- WBUS<23:20> 23=HALT PENDING 22=CLEAR HALT 21=CLR XMIT INT REQ 20=SET HALT
WCTRL/CONREAD	WBUS<23:16> <- RBUF (CLEARS DONE AND RECEIVER INT REQ)	WBUS<23:22> <- XCSR 23=READY 22=INT ENABLE	WBUS<23:22> <- RCSR 23=DONE 22=INT ENABLE	WBUS<23:21> <- CSR 23=HALT PENDING 22=HALT 21=0
WCTRL/READCRAR	WBUS<25:24> <- CRAR			

```
;2218   .TOC "  Micro Level Charts              . TABLE 3 - TU58 Interface Control"
;2219
;2220
;2221   ;       TRAR    : TU58 'Register Address' Register
;2222   ;
;2223   ;       XBUF    : Xmitter data BUFfer
;2224   ;       RBUF    : Reciever data BUFfer
;2225   ;       XCSR    : Xmitter  Command and Status Register
;2226   ;       RCSR    : Reciever Command and Status Register
;2227   ;       CSR     : Command and Status Register
;2228   ;
;2229   ;
;2230   ;
;2231   ;
;2232   ;
;2233   ;
;2234   ;
;2235   ;
;2236   ;
;2237   ;
;2238   ;
;2239   ;
;2240   ;
;2241   ;
;2242   ;
;2243   ;
;2244   ;
;2245   ;
;2246   ;
;2247   ;
;2248   ;
;2249   ;
;2250   ;
;2251   ;
;2252   ;
;2253   ;
;2254   ;
;2255   ;
;2256   ;
;2257   ;
```

	TRAR=0 (DATA)	TRAR=1 (XMIT STATUS)	TRAR=2 (RECV STATUS)	TRAR=3 (COMMAND & STATUS)
WCTRL/LOADTRAR	TRAR <- WBUS<23:22>			
WCTRL/TU58WRITE	XBUF <- WBUS<23:16> (CLEARS READY)	XCSR<6> <- WBUS<22> INT ENABLE	RCSR<6> <- WBUS<22> INT ENABLE	CSR <- WBUS<23:20> 23=COLD START BIT 22=CLEAR START BREAK 21=CLR XMIT INT REQ 20=SET START BREAK
WCTRL/TU58READ	WBUS<23:16> <- RBUF (CLEARS DONE AND RECEIVER INT REQ)	WBUS<23:22> <- XCSR 23=READY 22=INT ENABLE	WBUS<23:22> <- RCSR 23=DONE 22=INT ENABLE	WBUS<23> <- COLD START BIT WBUS<21> <- XMIT INT REQ
WCTRL/READTRAR	WBUS<25:24> <- TRAR			

.TOC " Micro Level Charts : ROM layout map for control store"

Fields	PAR2, PAR1, LIT	MISC	SPW, MSRC	MSRC	ROT	ROT / ALPCTL	ALPCTL (MUX / ALU)	ALPCTL (ALU / DQ1)	BUT	BUT, DTYPE
Bits	79 78 77 76	75 74 73 72	71 70 69 68	67 66 65 64	63 62 61 60	59 58 57 56	55 54 53 52	51 50 49 48	47 46 45 44	43 42 41 40
0000-03FF	146	140	134	128	122	116	110	104	98	92
0400-07FF	145	139	133	127	121	115	109	103	97	91
0800-0BFF	144	138	132	126	120	114	108	102	96	90
0C00-0FFF	143	137	131	125	119	113	107	101	95	89
1000-13FF	142	136	130	124	118	112	106	100	94	88
1400-17FF	141	135	129	123	117	111	105	99	93	87

* note :
listing is inverted with respect to acctal prom pattern

Fields	LITRL / RSRC	LITRL / RSRC, ISTRM, CC	WCTRL	WCTRL, BUS	BUS, FPA	FPA	CLKX, JSR, NEXT	NEXT <- inverted ->	NEXT	NEXT*
Bits	39 38 37 36	35 34 33 32	31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 09 08	07 06 05 04	03 02 01 00
0000-03FF	86	80	74	68	62	56	50	44	38	32
0400-07FF	85	79	73	67	61	55	49	43	37	31
0800-0BFF	84	78	72	66	60	54	48	42	36	30
0C00-0FFF	83	77	71	65	59	53	47	41	35	29
1000-13FF	82	76	70	64	58	52	46	40	34	28
1400-17FF	81	75	69	63	57	51	45	39	33	27

;2312 .BIN

2512; This page intentionally left blank.

```
                                        ;2313       .TOC "REGION.MIC"
                                        ;2314       .TOC "REVISION 11.0"
                                        ;2315       ;       C. E. MCDOWELL
                                        ;2316
;2317       .NOBIN
;2318       .HEXADECIMAL
;2319       .TOC "  Revision History"
;2320
;2321       ; 11    Move CMODE boundary for rev 061.
;2322       ; 10    Add regions for G and H floating.
;2323       ;       Adjust IANDE region.
;2324       ; 09    Initial release.
```

```
;2325   .TOC "  Location of modules in control store"
;2326
;2327   ;0000-03FF      IRD1/   IRDX/   VIELD(part 2)
;2328
;2329   ;0400-07FF      INTLOG/ OSR,    FLOAT,  CONTRL, GHFLT(rom part)
;2330
;2331   ;0800-0BFF      INIT,   CONSOL, CHAR
;2332
;2333   ;0C00-0FFF      PCALL,  MISQUE, DECMAL(part of CVTLP & CVTPL),  MPRCHM, IANDE
;2334
;2335   ;1000-13FF      DECMAL(except CVTLP & CVTPL)
;2336
;2337   ;1400-17FF      CMODE,  EDIT/   MM/     UVERFY/ PRLDSV, VIELD(part 1)
;2338
;2339   ;2000-23FF      GHFLT(wcs part)
;2340
;2341   ;NOTE:  Files seperated by slashes are regioned into the same locations.
```

```
;2342   .TOC     ""      Control Store Region Expressions"
;2343
;2344   ;Initialize
;2345           .SET/INIT.R1L=0800
;2346           .SET/INIT.R1H=0882
;2347           .SET/INIT.R2L=0800
;2348           .SET/INIT.R2H=0882
;2349           .SET/INIT.R3L=0800
;2350           .SET/INIT.R3H=0882
;2351
;2352   ;Console
;2353           .SET/CONSOL.R1L=0883
;2354           .SET/CONSOL.R1H=0A37
;2355           .SET/CONSOL.R2L=0883
;2356           .SET/CONSOL.R2H=0A37
;2357           .SET/CONSOL.R3L=0883
;2358           .SET/CONSOL.R3H=0A37
;2359
;2360   ;Integer, Logical, and Address
;2361           .SET/INTLOG.R1L=0400
;2362           .SET/INTLOG.R1H=04F8
;2363           .SET/INTLOG.R2L=0400
;2364           .SET/INTLOG.R2H=04F8
;2365           .SET/INTLOG.R3L=0400
;2366           .SET/INTLOG.R3H=04F8
;2367
;2368   ;Floating Point and CRC
;2369           .SET/FLOAT.R1L=04F9
;2370           .SET/FLOAT.R1H=0721
;2371           .SET/FLOAT.R2L=04F9
;2372           .SET/FLOAT.R2H=0721
;2373           .SET/FLOAT.R3L=04F9
;2374           .SET/FLOAT.R3H=0721
;2375
;2376   ;Variable Length Bit Field
;2377           .SET/VIELD.R1L=17E2
;2378           .SET/VIELD.R1H=17EF
;2379           .SET/VIELD.R2L=0000
;2380           .SET/VIELD.R2H=03EA
;2381           .SET/VIELD.R3L=0000
;2382           .SET/VIELD.R3H=03EA
;2383
;2384   ;Control Instructions
;2385           .SET/CONTRL.R1L=0722
;2386           .SET/CONTRL.R1H=0775
;2387           .SET/CONTRL.R2L=0722
;2388           .SET/CONTRL.R2H=0775
;2389           .SET/CONTRL.R3L=0722
;2390           .SET/CONTRL.R3H=0775
;2391
;2392
;2393
;2394
;2395
;2396
```

```
;2397    ;Procedure Call
;2398            .SET/PCALL.R1L=0C00
;2399            .SET/PCALL.R1H=0C96
;2400            .SET/PCALL.R2L=0C00
;2401            .SET/PCALL.R2H=0C96
;2402            .SET/PCALL.R3L=0C00
;2403            .SET/PCALL.R3H=0C96
;2404
;2405    ;Miscellaneous and Queue
;2406            .SET/MISQUE.R1L=0C97
;2407            .SET/MISQUE.R1H=0D30
;2408            .SET/MISQUE.R2L=0C97
;2409            .SET/MISQUE.R2H=0D30
;2410            .SET/MISQUE.R3L=0C97
;2411            .SET/MISQUE.R3H=0D30
;2412
;2413    ;Character String
;2414            .SET/CHAR.R1L=0A38
;2415            .SET/CHAR.R1H=0BEB
;2416            .SET/CHAR.R2L=0A38
;2417            .SET/CHAR.R2H=0BEB
;2418            .SET/CHAR.R3L=0A38
;2419            .SET/CHAR.R3H=0BEB
;2420
;2421    ;Decimal String
;2422            .SET/DECMAL.R1L=1000
;2423            .SET/DECMAL.R1H=13F2
;2424            .SET/DECMAL.R2L=1000
;2425            .SET/DECMAL.R2H=13F2
;2426            .SET/DECMAL.R3L=0D31
;2427            .SET/DECMAL.R3H=0D9C
;2428
;2429    ;Edit
;2430            .SET/EDIT.R1L=14E0
;2431            .SET/EDIT.R1H=17E1
;2432            .SET/EDIT.R2L=14E0
;2433            .SET/EDIT.R2H=17E1
;2434            .SET/EDIT.R3L=14E0
;2435            .SET/EDIT.R3H=17E1
;2436
;2437    ;MTPR, MFPR, CHMX
;2438            .SET/MPRCHM.R1L=0D9D
;2439            .SET/MPRCHM.R1H=0ECB
;2440            .SET/MPRCHM.R2L=0D9D
;2441            .SET/MPRCHM.R2H=0ECB
;2442            .SET/MPRCHM.R3L=0D9D
;2443            .SET/MPRCHM.R3H=0ECB
;2444
;2445    ;PROBE, LDPCTX, SVPCTX
;2446            .SET/PRLDSV.R1L=14E0
;2447            .SET/PRLDSV.R1H=17E1
;2448            .SET/PRLDSV.R2L=14E0
;2449            .SET/PRLDSV.R2H=17E1
;2450            .SET/PRLDSV.R3L=14E0
;2451            .SET/PRLDSV.R3H=17E1
```

```
;2452
;2453    ;Interrupts and Exceptions
;2454            .SET/IANDE.R1L=0ECC
;2455            .SET/IANDE.R1H=0FFF
;2456            .SET/IANDE.R2L=0ECC
;2457            .SET/IANDE.R2H=0FFF
;2458            .SET/IANDE.R3L=0ECC
;2459            .SET/IANDE.R3H=0FFF
;2460
;2461    ;Memory Management
;2462            .SET/MM.R1L=14E0
;2463            .SET/MM.R1H=17E1
;2464            .SET/MM.R2L=14E0
;2465            .SET/MM.R2H=17E1
;2466            .SET/MM.R3L=14E0
;2467            .SET/MM.R3H=17E1
;2468
;2469    ;Compatibility Mode
;2470            .SET/CMODE.R1L=1400
;2471            .SET/CMODE.R1H=14DF
;2472            .SET/CMODE.R2L=1400
;2473            .SET/CMODE.R2H=14DF
;2474            .SET/CMODE.R3L=1400
;2475            .SET/CMODE.R3H=14DF
;2476
;2477    ;Micro Verify
;2478            .SET/MV.R1L=14E0
;2479            .SET/MV.R1H=17E1
;2480            .SET/MV.R2L=14E0
;2481            .SET/MV.R2H=17E1
;2482            .SET/MV.R3L=14E0
;2483            .SET/MV.R3H=17E1
;2484
;2485    ;Native Mode Operand Specifier
;2486            .SET/OSR.R1L=0400
;2487            .SET/OSR.R1H=04F8
;2488            .SET/OSR.R2L=0400
;2489            .SET/OSR.R2H=04F8
;2490            .SET/OSR.R3L=0400
;2491            .SET/OSR.R3H=04F8
;2492
;2493    ;IRD1
;2494            .SET/IRD1.R1L=0000
;2495            .SET/IRD1.R1H=03EA
;2496
;2497    ;IRDX
;2498            .SET/IRDX.R1L=0000
;2499            .SET/IRDX.R1H=03EA
;2500            .SET/IRDX.R2L=0000
;2501            .SET/IRDX.R2H=03EA
;2502
;2503
;2504
;2505
;2506
```

```
;2507    ;G and H floating (rom part)
;2508            .SET/GHFLT.R1L=0776
;2509            .SET/GHFLT.R1H=07EF
;2510            .SET/GHFLT.R2L=0776
;2511            .SET/GHFLT.R2H=07EF
;2512            .SET/GHFLT.R3L=0776
;2513            .SET/GHFLT.R3H=07EF
;2514
;2515    ;G and H floating (WCS part)
;2516            .SET/WCS.R1L=2000
;2517            .SET/WCS.R1H=23FF
;2518            .SET/WCS.R2L=2000
;2519            .SET/WCS.R2H=23FF
;2520            .SET/WCS.R3L=2000
;2521            .SET/WCS.R3H=23FF
;2522
                                        ;2523       .BIN
```

```
                                        ;2524      .TOC "DEFIN.MIC"
                                        ;2525      .TOC "REVISION 67.0"
                                        ;2526      ;    Jeff Peng, P. R. GUILBAULT
                                        ;2527
;2528    .NOBIN
;2529    .ULINE
;2530    .RTOL
;2531    .HEXADECIMAL
;2532    .SOURCE/35
;2533    .TITLE  "CLOKX Rev @@@@@, Clock rate = ???ns"
;2534    .SET/INIT=0              ;SWITCH THAT INDICATES INIT U-CODE FOR VALIDITY CHECK
;2535    .WIDTH/81
;2536    .NOCREF                  ;SET UP FOR CREF ONLY WHEN FULL ASSEMBLY
;2537    .FILLERPAGE
;2538    .BOUNDS/ROW 1:0000,03FF/ROW 2:0400,07FF/ROW 3:0800,0BFF
;2539    .BOUNDS/ROW 4:0C00,0FFF/ROW 5:1000,13FF/ROW 6:1400,17FF
;2540    .BOUNDS/WCS:2000,23FF/PCS:2800,2BFF
;2541
;2542    .TOC     "       Revision History"
;2543    ; 67     Add a VALIDITY check V025 to verify the hardware conflict of "MSRC/TEMP4" and "BUS/PROBE.WR".
;2544    ;          and correct some error comments.
;2545    ; 66     Increase the width of u-word by one; Add a PATCH field in bit #80; Move the VSIZE field to bit #84.
;2546    ; 65     ADD BRANCH ON FPA PRESENT
;2547    ; 64     Initial release.
```

```
        .TOC "  Comet Micro Word Chart"

        ;
        ;                                     +-----------------------------------------------------------------+
        ;                                     |                                                                 |
        ;                                     |                                               +-----------------+
        ;                                     |                     LONLIT                    |                 |
        ;                                     |                                               |      LITRL      |
        ;                                     |                                               |                 |
        ;         +-------------+-------------+-----------------------------------------------+-----------+-+---+---------------------+
        ;         |      2      |                                  1                                      |2| 1 |          2          |
        ;   +-+-+-+---+---------+---+---------+-----------+-------------------+-----------+---+-----------+-+---+-----------+---------+
        ;   |P| | |   |         |   |         |           |                   |           | D |           |I|   |           |         |
        ;   |A|P|P|   |         |   |         |           |                   |           | T |           |S|   |           |         |
        ;   |T|A|A|LIT|  MISC   |SPW|  MSRC   |    ROT    |      ALPCTL       |    BUT    | Y |   RSRC    |T| C |   WCTRL   |   BUS   |
        ;   |C|R|R|   |         |   |         |           |                   |           | P |           |R| C |           |         |
        ;   |H|2|1|   |         |   |         |           |                   |           | E |           |M|   |           |         |
        ;   +-+-+-+---+---------+---+---------+-----------+-------------------+-----------+---+-----------+-+---+-----------+---------+
        ;   |8|7|7|7 7|7 7 7 7 7|7 6|6 6 6 6 6|6 6 6 6 5 5|5 5 5 5 5 5 5 5 4 4|4 4 4 4 4 4|4 4|3 3 3 3 3 3|3|3 3|3 2 2 2 2 2|2 2 2 2 2|
        ;   |0|9|8|7 6|5 4 3 2 1|0 9|8 7 6 5 4|3 2 1 0 9 8|7 6 5 4 3 2 1 0 9 8|7 6 5 4 3 2|1 0|9 8 7 6 5 4|3|2 1|0 9 8 7 6 5|4 3 2 1 0|
        ;   +-+-+-+---+---------+---+---------+-+-----+---+-------+-------+---+-----------+---+-----------+-+---+-----------+---------+
        ;                                     | |     |   |       |       |   |                             |               |
        ;                                     |A|  A  | A |  MUX  |  ALU  |DQ1|                             |    CCMISC     |
        ;                                     |L|  L  | L |       |       |   |                             |               |
        ;                                     |U|  U  | U +-------+-------+---+                             +---+-----------+
        ;                                     |X|  S  | C |       |       |   |                                 |           |
        ;                                     |M|  H  | I |       | ALUOD |DQ2|                                 |   CCPSL   |
        ;                                     | |  F  |   |       |       |   |                                 |           |
        ;                                     +-+-----+---+       +-------+---+                                 +-----------+
        ;                                     |           |               |   |
        ;                                     |  ROTSRK   |               |DQ3|
        ;                                     |           |               |   |
        ;                                     +-----------+               +---+
        ;
        ;
        ;
        ;
        ;
        ;                                                 +---------------------------------------+
        ;                                                 |                   2                   |
        ;                                                 +-------+-+-+---------------------------+
        ;                                                 |       |C|J|                           |
        ;                                                 |  FPA  |L|S|           NEXT            |
        ;                                                 |       |K|R|                           |
        ;                                                 |       |X| |                           |
        ;                                                 +-------+-+-+---------------------------+
        ;                                                 |1 1 1 1|1|1|1 1 1 1 0 0 0 0 0 0 0 0 0 0|
        ;                                                 |9 8 7 6|5|4|3 2 1 0 9 8 7 6 5 4 3 2 1 0|
        ;                                                 +-------+-+-+---------------------------+
        ;
```

```
;2597   .TOC "  Machine Definition              : ALPCTL"
;2598
;2599   ALPCTL/=<57:48>,.DEFAULT=364            ;ALP SPECIAL FUNCTIONS                                                    ;ALU OPERATION FOR
;2600           NOP=364                                                 ;ALUOD/OR,MUX/Z.S,DQ1/NOP                         ;SETTING OF ALU FLAGS
;2601           WX_D_Q.Q_D=2D7                                          ;WMUX & D <- Q OLD      Q <- D OLD                      D+Q+CI.BCD
;2602           WX_D_Q.Q_M=0D7                                          ;WMUX & D <- Q OLD      Q <- MBUS                       M+Q+CI.BCD
;2603           WX_D_R.Q_D=257                                          ;WMUX & D <- RBUS       Q <- D OLD                      D+R+CI.BCD
;2604           WX_D_R.Q_M=057                                          ;WMUX & D <- RBUS       Q <- MBUS                       M+R+CI.BCD
;2605           WX_D_R.Q_XM=157,        .VALIDITY=<V070>                ;WMUX & D <- RBUS       Q <- S/Z MBUS                  XM+R+CI.BCD
;2606           WX_D_S.Q_0=357                                          ;WMUX & D <- SUP ROT    Q <- 0                          0+S+ 0.BCD
;2607           WX_D_S.Q_R=3D7                                          ;WMUX & D <- SUP ROT    Q <- RBUS                       R+S+ 0.BCD
;2608           WX_D_S.Q_XM=1D7,        .VALIDITY=<V070>                ;WMUX & D <- SUP ROT    Q <- S/Z MBUS                  XM+S+ 0.BCD
;2609           WX_Q.Q_D=2C7                                            ;WMUX     <- Q OLD      Q <- D                          D-Q-CI.BCD
;2610           WX_Q.Q_M=0C7                                            ;WMUX     <- Q OLD      Q <- MBUS                       M-Q-CI.BCD
;2611           WX_R.Q_D=247                                            ;WMUX     <- RBUS       Q <- D                          D-R-CI.BCD
;2612           WX_R.Q_M=047                                            ;WMUX     <- RBUS       Q <- MBUS                       M-R-CI.BCD
;2613           WX_R.Q_XM=147,          .VALIDITY=<V070>                ;WMUX     <- RBUS       Q <- S/Z MBUS                  XM-R-CI.BCD
;2614           WX_S.Q_0=347                                            ;WMUX     <- SUP ROT    Q <- 0                          0-S- 0.BCD
;2615           WX_S.Q_R=3C7                                            ;WMUX     <- SUP ROT    Q <- RBUS                       R-S- 0.BCD
;2616           WX_S.Q_XM=1C7,          .VALIDITY=<V070>                ;WMUX     <- SUP ROT    Q <- S/Z MBUS                  XM-S- 0.BCD
;2617
;2618           WX_D_Q_S=373                                            ;WMUX & D & Q <- SUP ROT
;2619           WX_D_S=372                                              ;WMUX & D     <- SUP ROT
;2620           WX_Q_S=371                                              ;WMUX & Q     <- SUP ROT
;2621           WX_S=370                                                ;WMUX         <- SUP ROT
;2622           WX_D_Q_.NOT.S=363                                       ;WMUX & D & Q <- .NOT.(SUP ROT)
;2623           WX_D_.NOT.S=362                                         ;WMUX & D     <- .NOT.(SUP ROT)
;2624           WX_Q_.NOT.S=361                                         ;WMUX & Q     <- .NOT.(SUP ROT)
;2625           WX_.NOT.S=360                                           ;WMUX         <- .NOT.(SUP ROT)
;2626
;2627           WX_D_DSL.SQL=24E                                        ;WMXU & D <- D SHF LEFT     Q <- SHF LEFT       (D-R-CI).SL
;2628           WX_D_DSL.SQR=24F                                        ;WMXU & D <- D SHF LEFT     Q <- SHF RIGHT      (D-R-CI).SL
;2629           WX_D_DSR.SQL=24A                                        ;WMXU & D <- D SHF RIGHT    Q <- SHF LEFT       (D-R-CI).SR
;2630           WX_D_DSR.SQR=24B                                        ;WMXU & D <- D SHF RIGHT    Q <- SHF RIGHT      (D-R-CI).SR
;2631
;2632           WB_LOOPF=378                                            ;WB<31:30> <- 0'LOOP FLAG
;2633           WB_LOOPF.Q_0=379                                        ;WB<31:30> <- 0'LOOP FLAG   Q     <- 0
;2634           WB_LOOPF.D_0=37A                                        ;WB<31:30> <- 0'LOOP FLAG   D     <- 0
;2635           WB_LOOPF.Q_D_0=37B                                      ;WB<31:30> <- 0'LOOP FLAG   Q & D <- 0
;2636           WB_ALUF=37C                                             ;WB<31:30> <- ALUSO'ALKC
;2637           WB_ALUF.Q_S=37D                                         ;WB<31:30> <- ALUSO'ALKC    Q     <- S
;2638           WB_ALUF.D_S=37E                                         ;WB<31:30> <- ALUSO'ALKC    D     <- S
;2639           WB_ALUF.Q_D_S=37F                                       ;WB<31:30> <- ALUSO'ALKC    Q &  D <- S
;2640
;2641           MULFAST+=279,    .VALIDITY=<V21.50-54.70>               ;MULTIPLY +RBUS BY Q (2 ITERATIONS PER CYCLE)
;2642           MULSLOW+=278,    .VALIDITY=<V50.51.70>                  ;MULTIPLY +RBUS BY Q (1 ITERATION  PER CYCLE)
;2643           MULFAST-=269,    .VALIDITY=<V21.50-54.70>               ;MULTIPLY -RBUS BY Q (2 ITERATIONS PER CYCLE)
;2644           MULSLOW-=268,    .VALIDITY=<V50.51.70>                  ;MULTIPLY -RBUS BY Q (1 ITERATION  PER CYCLE)
;2645           DIVFAST+=26C,    .VALIDITY=<V21.50-54.70>               ;DIVIDE Q BY +RBUS (2 ITERATIONS PER CYCLE)
;2646           DIVSLOW+=26E,    .VALIDITY=<V50.51.70>                  ;DIVIDE Q BY +RBUS (1 ITERATION  PER CYCLE)
;2647           DIVFAST-=27C,    .VALIDITY=<V21.50-54.70>               ;DIVIDE Q BY -RBUS (2 ITERATIONS PER CYCLE)
;2648           DIVSLOW-=27E,    .VALIDITY=<V50.51.70>                  ;DIVIDE Q BY -RBUS (1 ITERATION  PER CYCLE)
;2649           REM=26A,         .VALIDITY=<V050>                       ;UNSHIFT REMAINDER (RBUS MUST BE 0)
;2650           DIVDA=27F,       .VALIDITY=<V050>                       ;DIVIDE DOUBLE ADD
;2651           DIVDS=26F,       .VALIDITY=<V050>                       ;DIVIDE DOUBLE SUB
```

```
;2652    .TOC "  Machine Definition              : ALU, ALUCI, ALUOD, ALUSHF, ALUXM"
;2653
;2654    ALU/=<53:50>            ;ALU FUNCTION CONTROL FOR MUX/(M.R1, M.R2, M.Q1, M.Q2,
;2655                            ;M.S, XM.R, XM.Q, ZM.S, D.R1, D.Q1, D.Q2, D.S, R.Q, R.S)
;2656            A-B-CI=0        ;SUB WITH CARRY INPUT
;2657            A-B-CI.BCD=1    ;BCD SUB WITH CARRY INPUT
;2658            A-B-CI.SR=2     ;(SUB WITH CARRY INPUT) SHIFTED RIGHT
;2659            A-B-CI.SL=3     ;(SUB WITH CARRY INPUT) SHIFTED LEFT
;2660            B-A-CI=0C       ;SUB WITH CARRY INPUT
;2661
;2662            A+B+CI=4        ;ADD WITH CARRY INPUT
;2663            A+B+CI.BCD=5    ;BCD ADD WITH CARRY INPUT
;2664            A+B+CI.SR=6     ;(ADD WITH CARRY INPUT) SHIFTED RIGHT
;2665            A+B+CI.SL=7     ;(ADD WITH CARRY INPUT) SHIFTED LEFT
;2666
;2667            AND=8           ;A.AND.B
;2668            OR=9            ;A.OR.B
;2669            AND.SR=0A       ;(A.AND.B) SHIFTED RIGHT
;2670            AND.SL=0B       ;(A.AND.B) SHIFTED LEFT
;2671            XOR=0D          ;A.XOR.B
;2672            ANDNOT=0E       ;A.AND.(.NOT.B)
;2673            NOTAND=0F       ;(.NOT.A).AND.B
;2674
;2675    ALUCI/=<59:58>          ;ALU CARRY INPUT CONTROL WHEN MUX DOES NOT SELECT S ROT AND P OR S LATCH IS NOT MODIFIED
;2676            ZERO=0          ;CI <- 0
;2677            ALKC=1          ;CI <- ALK<C>
;2678            ONE=2           ;CI <- 1
;2679            PSLC=3          ;CI <- PSL<C>
;2680
;2681    ALUOD/=<53:50>          ;ALU FUNCTION CONTROL FOR MUX/(D.R2, Z.S)
;2682            AND.OD=8        ;A.AND.B                    OUTPUT DISABLE
;2683            OR.OD=9         ;A.OR.B                     OUTPUT DISABLE
;2684            AND.SR.OD=0A    ;(A.AND.B) SHFTD RIGHT      OUTPUT DISABLE
;2685            AND.SL.OD=0B    ;(A.AND.B) SHFTD LEFT       OUTPUT DISABLE
;2686            B-A-CI.OD=0C    ;SUB WITH CARRY             OUTPUT DISABLE
;2687            XOR.OD=0D       ;A.XOR.B                    OUTPUT DISABLE
;2688            ANDNOT.OD=0E    ;A.AND.(.NOT.B)             OUTPUT DISABLE
;2689            NOTAND.OD=0F    ;(.NOT.A).AND.B             OUTPUT DISABLE
;2690
;2691    ALUSHF/=<62:60>         ;ALP SHIFT-IN CONTROL WHEN MUX DOES NOT SELECT S ROT AND P OR S LATCH IS NOT MODIFIED
;2692                            ;ALU      Q
;2693            ;HARD DEFAULT   ; 0        0
;2694            ZERO=0          ; 0        0
;2695            ONE=1           ; 1        1
;2696            SHF=2           ;(SHIFT ALU'Q)  (SEE TABLE 1)
;2697            ROT=3           ;(ROT   ALU'Q)  (SEE TABLE 1)
;2698            ALU0.Q1=4       ; 0        1
;2699            ALU1.Q0=5       ; 1        0
;2700            WBUS30=6        ;WBUS<30> WBUS<30>
;2701            PSLC=7          ;PSL<C>   PSL<C>
;2702
;2703    ALUXM/=<63:63>          ;MBUS SIGN/ZERO EXT CONTOL
;2704            ZERO=0          ;ZERO EXTEND MBUS
;2705            SIGN=1          ;SIGN EXTEND MBUS
```

```
;2706   .TOC "  Machine Definition                    : BUS"
;2707
;2708   BUS/=<24:20>,.DEFAULT=7                      ;BUS FUNCTIONS
;2709           NOP=7                                   ;NO OP
;2710           READ=10,          .VALIDITY=<V22.70>    ;MDR <- MEMORY PER BY VA AND DSIZE.
;2711           READ.MOD=14,      .VALIDITY=<V22.70>    ;CHECK FOR WRITE ACCESS THEN "RD".
;2712           READ.LNG=11,      .VALIDITY=<V022>      ;READ IGNORING THE TWO LSB'S OF THE VA
;2713           READ.LNG.MOD=15,.VALIDITY=<V022>        ;CHECK FOR WRITE ACCESS THE "RD.L".
;2714           READ.NT=2,        .VALIDITY=<V022>      ;SUPPRESS ACV AND UNALIGNED DATA U-TRAPS
;2715           READ.MOD.LCK=13,.VALIDITY=<V22.70>      ;READ, LOCK OUT OTHER READ LOCKS
;2716           READ.PHY=0,       .VALIDITY=<V022>      ;READ TREATING VA AS PHYSICAL ADDRESS, IGNORE 2 LSB'S.
;2717           READ.SEC=6,       .VALIDITY=<V022>      ;FETCH REMAINDER OF A READ CROSSING LONG BOUNDARY.
;2718
;2719           WRITE=18,         .VALIDITY=<V20.70>    ;WRITE MEMORY SPECIFIED BY VA, DSIZE, WDR
;2720           WRITE.NOREG=1A,   .VALIDITY=<V20.70>    ;WRITE UNLESS REGISTER MODE ASSERTTED.
;2721           WRITE.LNG=19,     .VALIDITY=<V20.70>    ;WRITE IGNORING THE 2 LSB'S OF THE VA
;2722           WRITE.NT=0C,      .VALIDITY=<V020>      ;WRITE. SUPPRESS ACV, UNALIGNED DATA, AND PAGE BOUNDARY U-TRAPS
;2723           WRITE.NT.LNG=0E,.VALIDITY=<V020>        ;"WRITE.NOTRP" LONGWORD. USED FOR WRITING THE M-BIT.
;2724           WRITE.UL=1B,      .VALIDITY=<V20.70>    ;WRITE. RELEASE LOCK SET BY READ.LCK
;2725           WRITE.PHY=8,      .VALIDITY=<V020>      ;WRITE TREATING VA AS PHYSICAL ADDRESS, IGNORE 2 LSB'S.
;2726           WRITE.SEC=0A,     .VALIDITY=<V020>      ;WRITE SECOND REFERENCE
;2727           WRITE.UL.SEC=0B,.VALIDITY=<V020>        ;WITER.SEC. RELEASE LOCK SET BY "RD.LK".
;2728
;2729           PRB.RD=1F,        .VALIDITY=<V24.25.70> ;PROBE PTE IN TB FOR READ ACCESS AT ADDRESS IN VA.
;2730           PRB.RD.MODE=1E,.VALIDITY=<V24.25.70>    ;PRB.R USING WBUS<25:24> INSTEAD OF THE CURRENT MODE.
;2731           PRB.RD.PTE=16,    .VALIDITY=<V023>      ;PROBE PTE IMAGE ON WBUS FOR READ ACCESS.
;2732           PRB.RD.PTE.K=17,.VALIDITY=<V023>        ;PTE.R USING KERNEL MODE INSTEAD OF CURRENT MODE.
;2733           PRB.WR=1D,        .VALIDITY=<V24.25.70> ;PROBE PTE IN TB FOR WRITE ACCESS AT ADDRESS IN VA.
;2734           PRB.WR.MODE=1C,.VALIDITY=<V24.25.70>    ;PRB.W USING WBUS<25:24> INSTEAD OF THE CURRENT MODE.
;2735           PRB.WR.PTE=12,    .VALIDITY=<V023>      ;PROBE PTE IMAGE ON WBUS FOR WRITE ACCESS.
;2736           IOINIT=3,         .VALIDITY=<V021>      ;GENERATE UNIBUS INIT.
;2737           PRINIT=1,         .VALIDITY=<V021>      ;GENERATE A MASTER PROCESSOR RESET. (INCLUDES AN IOINIT)
;2738           REICHK=9,         .VALIDITY=<V008>      ;REI CHECK
;2739           GRANT=0F,         .VALIDITY=<V09.21>    ;ISSUE UNIBUS GRANT
```

```
;2740    .TOC "  Machine Definition              : BUT"
;2741
;2742    BUT/=<47:42>,.DEFAULT=0
;2743                            ; NOTE 01 : ALU/A+B+CI.BCD, ALEG/DATA, BLEG/0.  A 1 IN CSA<0> Indicates Illegal BCD Char
;2744
;2745                            ; NOTE 02 : VA<0>.AND.NOT.IR<3> FOR Native Mode, VA<0>.AND.NOT.IR<7> FOR Compatibility Mode
;2746
;2747                            ; NOTE 03 : STEPC <- STEPC - 4 & Branch ON Original Value  CSA<2>  CSA<1>  CSA<0>
;2748                            ;           ORIG VALUE = 0                                    0       0       0
;2749                            ;           ORIG VALUE = 1                                    0       0       1
;2750                            ;           ORIG VALUE = 2                                    0       1       0
;2751                            ;           ORIG VALUE = 3                                    0       1       1
;2752                            ;           ORIG VALUE >OR= 4, AND NO INT OR TIM OV           1       0       0
;2753                            ;           ORIG VALUE >OR= 4, AND INT OR TIM OV              1       0       1
;2754
;2755                            ; NOTE 04 : CSA<1> = (.NOT.MM.NOINT).AND.(INT.OR.TIMSERV)
;2756
;2757                            ; NOTE 05 : For Description of BUT/SPASTA Operation See Tables and Charts
;2758
;2759                            ; NOTE 06 : For Description of BUT/SRKSTA Operation See ROT & ROTSRK Micro field Definitions
;2760
;2761                            ; NOTE 07 : For Description of BUT/CCBR Operation See Tables and Charts
;2762
;2763                            ; NOTE 08 : In Compatibility Mode BUT is on CIR(See MSQ Spec Page 32)
;2764
;2765                            ; NOTE 09 : For Description of BUT/UVCTR Operation See Tables and Charts
;2766
;2767                            ; NOTE 10 :  START<1>  START<0>  Function
;2768                            ;              0         0      Restart/Boot
;2769                            ;              0         1      Restart/Halt
;2770                            ;              1         0      Boot
;2771                            ;              1         1      Halt
;2772
;2773                            ; NOTE 11 :  BOOT<1>   BOOT<0>   Function
;2774                            ;              0         0         D
;2775                            ;              0         1         C
;2776                            ;              1         0         B
;2777                            ;              1         1         A
;2778
;2779            NOP=0           ;NO BRANCH
;2780            IRD1=4          ;EVALUATE OPCODE AND 1ST OPERAND OF NEXT INSTRUCTION
;2781            IRD1TST=5       ;SAME AS IRD1 EXCEPT NEXT ADDRESS IS FROM 'NEXT'
;2782            IRDX=1          ;RETURN FROM OPERAND EVALUATION
;2783            RETURN=2        ;POP U-STK AND RET TO 'STK + NEXT'(MOD64)
;2784            RET.DINH=3      ;POP U-STK AND RET TO 'STK + NEXT'(MOD64),DEST INH
;2785            LOD.INC.BRA=6   ;LOAD OSR, INC IRDCNT, BRANCH ON ADD MODE
;2786            LOD.BRA=7       ;LOAD OSR,           , BRANCH ON ADD MODE
;2787            BRA.ON.ADD=18   ;       ,            , BRANCH ON ADD MODE
```

	CSA<5>	CSA<4>	CSA<3>	CSA<2>	CSA<1>	CSA<0>
BUTXB=20						BUT XB UTRAP = 1, ELSE 0
CM.CUD.ADD=21						NOTE 2
IR.2TO0=19				IR<2>	IR<1>	IR<0>
IR5=23						IR<5> (NOTE 8)
IR2=22						IR<2> (NOTE 8)
REGMODE=24						OSR<7:4>=5
FRO.FLTZ=2A					NOT.MBUS<15>	WX<31:0>.NE.0
WBUS1TO0=9					WBUS<1>	WBUS<0>
WBUS1TO0.NE.0=0E						WBUS<1:0>,NE.0
WBUS5TO0=08	WB<5>	WB<4>	WB<3>	WB<2>	WB<1>	WB<0>
WBUS0=0A						WB<0>
WBUS31TO30=1B					WB<31>	WB<30>
WX.EQ.0=28						WMUX<31:0>.EQ.0
WX.NE.0=29						WMUX<31:0>.NE.0
BCDCHK=26						NOTE 1
SRKSTA=37					SRKSTA<1>	SRKSTA<0> (NOTE 6)
SPASTA=2E					SPASTA<1>	SPASTA<0> (NOTE 5)
CCBR=2D					CCBR<1>	CCBR<0> (NOTE 7)
CCBR1.CCBR0.IR0=35				CCBR<1>	CCBR<0>	.NOT.IR<0> (NOTE 8)
CCBR0.SRKSTA0=36					CCBR<0>	SRKSTA<0>
DSIZE=31					DSIZE<1>	DSIZE<0> (LATCHES)
DBZ.SC-0C						SC.EQ.0
NO.FPA=30						0=FPA PRESENT, 1=NO FPA
WCSENA=27						0=WCS PRESENT AND ENABLED
						1=WCS NOT PRESENT OR WCS DISABLED
BR.SC-4.INT-TS=0D				NOTE 3	NOTE 3	NOTE 3
MM.ALLOW.INT=0B					NOTE 4	
INT-TIMSERV=2B					.NOT.INT	TIMSERV
CCBR1.INT-TS=2C					CCBR<1>	(INT.OR.TIMSERV).AND.NOT.CCBR<1>
FPD=0F						PSL<27>
PSLC=25						PSL<C>
PSLTP=2F						PSL<TP>
UVCTR=1E			UVCTR<3>	UVCTR<2>	UVCTR<1>	UVCTR<0> (NOTE 9)
FPS1=34				NOT.HALT	START<1>	START<0> (NOTE 10)
FPS2=33					BOOT<1>	BOOT<0> (NOTE 11)
FPS3=32					ACLO	FP_OCK
FLAG0=10						FLAG0
FLAG1=17					FLAG1	
FLAG2=12				FLAG2		
FLAG3=13						FLAG3
MM.NOINT=11						MM.NOINT (FLAG4)
STACKFLG=1A						PSHSTACK (FLAG5)
FLAG2TO0=16				FLAG2	FLAG1	FLAG0
FLAG1TO0=14					FLAG1	FLAG0
F1.XOR23=15					FLAG1	FLAG2.XOR.FLAG3

```
;2842    .TOC "  Machine Definition              : CC, CCMISC, CCPSL"
;2843
;2844
;2845    CC/=<32:31>,.DEFAULT=0                                          ;CC CONTROL                 CCBR CONTROL
;2846            NOP.CCBR_SIGND=0,               .VALIDITY=<V071>        ;NOP                        SIGNED COMPARE
;2847            NOP.CCBR_ALUS=3                                         ;NOP                        ALU STATE LATCHES
;2848            CCOP1.CCBR_SIGND=1,             .VALIDITY=<V070>        ;CC OP 1                    SIGNED COMPARE
;2849            CCOP2.CCBR_SIGND=2,             .VALIDITY=<V070>        ;CC OP 2                    SIGNED COMPARE
;2850
;2851
;2852
;2853
;2854
;2855    CCMISC/=<32:25>                                                 ;MISC                        CCBR CONTROL
;2856            NOP.CCBR_BRATST=0C7                                     ;NOP                        BRANCH CONDITION TRUE
;2857            NOP.CCBR_CSIGNS=46,             .VALIDITY=<V071>        ;NOP                        COMPARE SIGNS
;2858            WB_ATCR.CCBR_SIGND=7,           .VALIDITY=<V71.80>      ;WBUS<3:0> <- ATCR          SIGNED COMPARE
;2859            ALUS_DSDZ.CCBR_ALUS=6                                   ;ALUS <- DEC SIGN DEC 0     OLD ALUS
;2860            ALUS_SIGND.CCBR_ALUS=0C6,       .VALIDITY=<V070>        ;ALUS <- SIGNED COMPARE     OLD ALUS
;2861            ALUS_UNSGN.CCBR_ALUS=86,        .VALIDITY=<V070>        ;ALUS <- UNSIGNED COMPARE   OLD ALUS
;2862            SETV.CCBR_SIGND=47,             .VALIDITY=<V071>        ;PSL<V> <- 1                SIGNED COMPARE
;2863
;2864
;2865
;2866
;2867
;2868    CCPSL/=<30:25>                                                  ;PSL(W)                     CCBR CONTROL
;2869            WB_PSL.CCBR_SIGND=4,            .VALIDITY=<V00.71.80>   ;READ PSL                   SIGNED COMPARE
;2870            CC_WB.CCBR_ALUS=5                                       ;CC <- WBUS<3:0>            ALU STATE LATCHES
;2871            PSL_WB.CCBR_ALUS=0,             .VALIDITY=<V001>        ;WRITE PSL                  ALU STATE LATCHES
;2872            PSW_WB.CCBR_ALUS=1                                      ;WRITE PSW                  ALU STATE LATCHES
;2873            MDR_OSR.CCBR_BRATST=2F,         .VALIDITY=<V02.33.41>   ;MDR <- ZEXT OSR            BRANCH CONDITION TRUE
;2874
;2875    ; NOTE : DEFAULT FOR CC & CCMISC DURING LONG OR SHORT LITERAL IS 'CC/NOP.CCBR_SIGND'.
;2876    ;
;2877    ; NOTE : FOR DESCRIPTION OF BUT/CCBR SEE TABLES AND CHARTS.
```

```
;2878    .TOC "  Machine Definition              : CLKX, DQ1, DQ2, DQ3, DTYPE, FPA"
;2879
;2880    CLKX/=<15:15>,.DEFAULT=0        ;CLOCK EXTEND CONTROL
;2881            NOP=0                                   ;NORMAL CLOCK
;2882            XTND=1                                  ;STRETCH CYCLE
;2883
;2884    DQ1/=<49:48>,.DEFAULT=0         ;D&Q REG CONT FOR MUX/(M.R1, M.Q1, M.S, XM.R, XM.Q, XM.S, D.R1, D.Q1, D.S, Z.S, R.Q, R.S)
;2885            NOP=0                                   ;NOP
;2886            Q_WX=1                                  ;Q <- WMUX
;2887            D_WX=2                                  ;D <- WMUX
;2888            Q_D_WX=3                                ;Q <- WMUX      D <- WMUX
;2889
;2890    DQ2/=<49:48>                    ;D&Q REG CONT FOR MUX/(M.R2, M.Q2, D.Q2)
;2891            SQL=0                                   ;SHF Q LEFT
;2892            SQR=1                                   ;SHF Q RIGHT
;2893            SQL.D_WX=2                              ;SHF Q LEFT     D <- WMUX
;2894            SQR.D_WX=3                              ;SHF Q RIGHT    D <- WMUX
;2895
;2896    DQ3/=<49:48>                    ;D&Q REG CONT FOR MUX/D.R2
;2897            SQL.D_WX=0                              ;SHF Q LEFT     D <- WMUX
;2898            SQR.D_WX=1                              ;SHF Q RIGHT    D <- WMUX
;2899
;2900    DTYPE/=<41:40>,.DEFAULT=3
;2901            BYTE=0                                  ;DATA SIZE IS BYTE
;2902            WORD=1                                  ;DATA SIZE IS WORD
;2903            LONG=2                                  ;DATA SIZE IS LONG WORD
;2904            IDEP=3                                  ;DATA SIZE IS INSTRUCTION DEPENDANT
;2905
;2906    FPA/=<19:16>,.DEFAULT=0         ;FLOATING POINT ACCELERATOR CONTROL
;2907            NOP=0                                   ;
;2908
;2909            FPA_DATA.MBUS=1                         ;FPA <- MBUS (Use when not 2nd half of dble OS2 REG dest) *See note*
;2910            FPA_DATA.WBUS=6                         ;FPA <- WBUS (Use when     2nd half of dble OS2 REG dest) *See note*
;2911            FPA_MBUS.LITNXT=2                       ;FPA <- MBUS, NEXT XFER WILL BE SHORT LITERAL
;2912            FPA_MBUS.FPA_WBUS=3                     ;FPA <- BOTH MBUS & WBUS
;2913            WBUS_FPA=4,             .VALIDITY=<V080>        ;WBUS <- FPA DATA
;2914            WBUS_FPA.CC=5,          .VALIDITY=<V080>        ;WBUS<3:0> <- CC'S FROM FPA
;2915
;2916    ; Note : The presence of either micro-order(6 or 1) instructs the FPA to accept data. The FPA does not distinguish
;2917    ;        between the two micro-orders and automaticly accepts data from the correct buss. The different micro-orders
;2918    ;        exist to enable the CLOKX post processor to determine from which buss the FPA is recieving data.
```

```
;2919    .TOC "  Machine Definition                : ISTRM, JSR, LIT, LITRL, LONLIT, MISC"
;2920
;2921    ISTRM/=<33:33>,.DEFAULT=0
;2922            NOP=0                                   ;ISIZE IS DETERMINED BY HARDWARE
;2923            ISIZE_DSIZE=1,  .VALIDITY=<V070>        ;ISIZE IS DETERMINED BY DSIZE
;2924
;2925    JSR/=<14:14>,.DEFAULT=0                         ;SUBROUTINE CONTROL
;2926            NOP=0                                   ;NO OPERATION
;2927            PUSH=1                                  ;PUSH CURRENT ADDRESS ON MICRO STACK
;2928
;2929    LIT/=<77:76>,.DEFAULT=0                         ;DEFINE UWORD FIELD INTERPRETATIONS
;2930            NORMAL=0                                ;FIELDS ARE NORMAL
;2931            LITRL=1,        .VALIDITY=<V071>        ;SHORT LITERAL FIELD ENABLED
;2932            FPAWAIT=2                               ;WAIT FOR FPA TO COMPLETE PROCESSING
;2933            LONLIT=3                                ;LONG LITERAL FIELD ENABLED
;2934
;2935    LITRL/=<39:31>,         .VALIDITY=<V071>        ;SHORT LITERAL
;2936
;2937    LONLIT/=<62:31>                                 ;LONG LITERAL
;2938
;2939    MISC/=<75:71>,.DEFAULT=10                       ;DEFINE MISC FUNCTIONS
;2940            NOP=10
;2941
;2942            CLR.FLAG0=0                             ;CLEAR FLAG 0
;2943            CLR.FLAG1=1                             ;CLEAR FLAG 1
;2944            CLR.FLAG2=2                             ;CLEAR FLAG 2
;2945            CLR.FLAG3=3                             ;CLEAR FLAG 3
;2946            CLR.MMNOINT=4                           ;CLEAR FLAG 4
;2947            CLR.STACKFLG=5                          ;CLEAR FLAG 5
;2948
;2949            SET.FLAG0=8                             ;SET FLAG 0
;2950            SET.FLAG1=9                             ;SET FLAG 1
;2951            SET.FLAG2=0A                            ;SET FLAG 2
;2952            SET.FLAG3=0B                            ;SET FLAG 3
;2953            SET.MMNOINT=0C                          ;SET FLAG 4
;2954            SET.STACKFLG=0D                         ;SET FLAG 5
;2955
;2956            RSBC=1B                                 ;RETURN AND SUPPRESS BUS CYCLE
;2957            RNUM_2REG=11                            ;RNUM <- COMP MODE SECOND REG
;2958            CLR.TP=12                               ;PSL<TP> <- 0
;2959            CLR.FPD=1C                              ;PSL<FPD> <- 0
;2960            SET.FPD=1D                              ;PSL<FPD> <- 1
;2961            FORCE.TB=1E                             ;FORCE TB PARITY ERROR
;2962            FORCE.CACHE=1F                          ;FORCE CACHE PARITY ERROR
;2963
;2964            DEC.SC=13                               ;STEP CNT <- STEP CNT - 1
;2965            SC_2=14                                 ;STEP CNT <- 2
;2966            SC_6=15                                 ;STEP CNT <- 6
;2967            SC_14=16                                ;STEP CNT <- 14
;2968            SC_30=17                                ;STEP CNT <- 30
```

```
;2969    .TOC "  Machine Definition                : MSRC"
;2970
;2971    MSRC/=<68:64>,.DEFAULT=0;M SCRATCH PAD ADD, M BUS, RBS, AND RNUM CONTROL
;2972
;2973            TEMP0=00                                ;MBUS <- DUAL PORT TEMP 0
;2974            TEMP1=01                                ;MBUS <- DUAL PORT TEMP 1
;2975            TEMP2=02                                ;MBUS <- DUAL PORT TEMP 2
;2976            TEMP3=03                                ;MBUS <- DUAL PORT TEMP 3
;2977            TEMP4=04                                ;MBUS <- DUAL PORT TEMP 4
;2978            TEMP5=05                                ;MBUS <- DUAL PORT TEMP 5
;2979            TEMP6=06                                ;MBUS <- DUAL PORT TEMP 6
;2980            TEMP7=07                                ;MBUS <- DUAL PORT TEMP 7
;2981            TEMP8=08                                ;MBUS <- M S-PAD TEMP 8
;2982            TEMP9=09                                ;MBUS <- M S-PAD TEMP 9
;2983            TEMP10=0A                               ;MBUS <- M S-PAD TEMP 10
;2984            ERRCOD=0B                               ;ERROR CODE FOR MEMORY FAULTS & ARITHMETIC TRAPS
;2985            FPDOFFSET=0C                            ;MBUS <- FPD PACK ROUTINE OFFSET
;2986            MM.TEMP0=0D                             ;MBUS <- MEMORY MANAG. TEMP 0
;2987            SCBB=0E                                 ;SYSTEM CONTROL BLOCK BASE
;2988            SISR=0F                                 ;SOFTWARE INT SUMMARY REGISTER
;2989
;2990            TEMP.R=10                               ;MBUS <- M S-PAD INDX BY RNUM
;2991            TEMP.R+1=11                             ;MBUS <- M S-PAD INDX BY RNUM+1
;2992
;2993                                                    ;RNUM CONTROL                                   DEFAULT MTMP
;2994            RNUM_WBUS=1D                            ;RNUM <- WBUS<3:0>                              MTMP0
;2995            WBUS_RNUM=16,    .VALIDITY=<V080>       ;WBUS<3:0> <- RNUM                              MTMP0
;2996
;2997                                                    ;RBS CONTROL                                    DEFAULT MTMP
;2998            PSHADD=15,       .VALIDITY=<V070>       ;PUSH !REG ADDRESS!DSIZE!+! & INC RBSP          MTMP0
;2999            PSHSUB=14,       .VALIDITY=<V070>       ;PUSH !REG ADDRESS!DSIZE!-! & INC RBSP          MTMP0
;3000            READRBS=1C,      .VALIDITY=<V080>       ;READ RBS           & INC RBSP                  MTMP0
;3001            WB_RBSP=1E,      .VALIDITY=<V080>       ;WBUS<3:0> <- 0'RBS POINTER                     MTMP0
;3002
;3003                                                    ;MEMORY INTERFACE CONTROL                       DEFAULT MTMP
;3004            MDR=12                                  ;MBUS <- MEMORY DATA REGISTER                   MTMP0  W ONLY
;3005            VA=1B                                   ;MBUS <- VIRTUAL ADDRESS REGISTER               MTMP0  W ONLY
;3006            PC=1A                                   ;MBUS <- PROGRAM COUNTER                        MTMP0  W ONLY
;3007            XB.PC_PC+I=17,   .VALIDITY=<V070>       ;MBUS <- EXECUTION BUF . PC <- PC+ISIZE         MTMP0  W ONLY
;3008            PCBACK=19                               ;MBUS <- PC BACK                                MTMP0  W ONLY
;3009            MA=18                                   ;MBUS <- MEMORY ADDRESS REGISTER                MTMP0  W ONLY
;3010            TB=1F                                   ;MBUS <- TRAN BUF DATA                          MTMP0  W ONLY
;3011            WDR=13,          .VALIDITY=<V038>       ;MBUS <- WRITE DATA REGISTER                    MTMP0  W ONLY
```

```
;3012    .TOC "  Machine Definition              : MUX, NEXT, PAR1, PAR2, PARF1A, PARF1B, PARF2"
;3013
;3014    MUX/=<57:54>                            ;ALP A AND B MUX INPUT CONTROL
;3015            M.R1=0                          ;A <- MBUS                B <- RBUS
;3016            M.R2=1                          ;A <- MBUS                B <- RBUS
;3017            M.Q1=2                          ;A <- MBUS                B <- Q REGISTER
;3018            M.Q2=3                          ;A <- MBUS                B <- Q REGISTER
;3019            M.S=4                           ;A <- MBUS                B <- SHIFTER
;3020            XM.R=5, .VALIDITY=<V070>        ;A <- EXTENDED MBUS       B <- RBUS
;3021            XM.Q=6, .VALIDITY=<V070>        ;A <- EXTENDED MBUS       B <- Q REGISTER
;3022            XM.S=7, .VALIDITY=<V070>        ;A <- EXTENDED MBUS       B <- SHIFTER
;3023            D.R1=8                          ;A <- D REGISTER          B <- RBUS
;3024            D.R2=9                          ;A <- D REGISTER          B <- RBUS
;3025            D.Q1=0A                         ;A <- D REGISTER          B <- Q REGISTER
;3026            D.Q2=0B                         ;A <- D REGISTER          B <- Q REGISTER
;3027            D.S=0C                          ;A <- D REGISTER          B <- SHIFTER
;3028            Z.S=0D                          ;A <- 0                   B <- SHIFTER
;3029            R.Q=0E                          ;A <- RBUS                B <- Q REGISTER
;3030            R.S=0F                          ;A <- RBUS                B <- SHIFTER
;3031    .CREF                                   ;END OF CREF ONLY WHEN FULL ASSEMBLY THE FOLLOWING WILL ALWAYS CREF
;3032
;3033    NEXT/=<13:0>,.NEXTADDRESS
;3034    PATCH/=<80:80>,.DEFAULT=0
```

```
;3035   .TOC "  Machine Definition                : ROT"
;3036
;3037   .NOCREF                                   ;SET UP FOR CREF ONLY WHEN FULL ASSEMBLY
;3038   ROT/=<63:58>                              ;SUPER ROTATOR CONTROL                                  | SRKSTA<1>        SRKSTA<0>
;3039                                            ;-------------------------------------------------------+-----------------------------
;3040           XZ.MR=0                          ;EXTRACT & ZERO EXTEND M'R, POS = PL, SIZE = SL         | SL.EQ.0          (PL<4:0>+SL).GT.32
;3041           XZ.MM=1                          ;EXTRACT & ZERO EXTEND M'M, POS = PL, SIZE = SL         | SL.EQ.0          (PL<4:0>+SL).GT.32
;3042           XZ.RR=2                          ;EXTRACT & ZERO EXTEND R'R, POS = PL, SIZE = SL         | SL.EQ.0          (PL<4:0>+SL).GT.32
;3043           XZ.VPN=0D,      .VALIDITY=<V072> ;EXTRACT & ZERO EXTEND M'M, POS = 09, SIZE = 21         | DSIZE<1>         DSIZE<0>
;3044           XZ.PTX=0C,      .VALIDITY=<V072> ;EXTRACT & ZERO EXTEND M'M, POS = 07, SIZE = 23         | DSIZE<1>         DSIZE<0>
;3045                                            ;                                                       |
;3046           CLR1BM=13                        ;CLR M<07:0>                                            | S<3:0>.NE.0      S<3:0>.NE.(11,13)
;3047           CLR2BM=12                        ;CLR M<15:0>                                            | S<3:0>.NE.0      S<3:0>.NE.(11,13)
;3048           CLR3BM=14                        ;CLR M<23:0>                                            | ASCII SIGN CHECK (SEE TABLE)
;3049                                            ;                                                       |
;3050           RL.RM.P=23                       ;ROT LEFT  R'M, NO. BITS = PLATCH<4:0>  (NOTE 1)        | WX<31:16>.NE.0 WX<15:0>.NE.0
;3051           RL.RM.PS=20                      ;ROT LEFT  R'M, NO. BITS = (PL+SL)<4:0> (NOTE 1)        | SL.EQ.0          UNDEFINED
;3052           RL.RM.4=8,      .VALIDITY=<V072> ;ROT LEFT  R'M, NO. BITS = 4                            | DSIZE<1>         DSIZE<0>
;3053           RL.MM.P=21                       ;ROT LEFT  M'M, NO. BITS = PLATCH                       | SL.EQ.0          PL<5>
;3054           RL.MM.PTE=11                     ;ROT LEFT  M'M, NO. BITS = 9                            | S<3:0>.NE.0      S<3:0>.NE.(11,13)
;3055           RL.RR.P=22                       ;ROT LEFT  R'R, NO. BITS = PLATCH                       | SL.EQ.0          PL<5>
;3056                                            ;                                                       |
;3057           RR.MR.P=4                        ;ROT RIGHT M'R, NO. BITS = PLATCH<4:0>                  | SL.EQ.0          (PL<4:0>+SL).GT.32
;3058           RR.MR.PS=24                      ;ROT RIGHT M'R, NO. BITS = (PL+SL)<4:0>                 | SL.EQ.0          (PL<4:0>+SL).GT.32
;3059           RR.MR.4=9,      .VALIDITY=<V072> ;ROT RIGHT M'R, NO. BITS = 4                            | DSIZE<1>         DSIZE<0>
;3060           RR.MR.S=7                        ;ROT RIGHT M'R, NO. BITS = SLATCH<4:0>                  |   0              PL<5>
;3061           RR.MR.9=0B,     .VALIDITY=<V072> ;ROT RIGHT M'R, NO. BITS = 9                            | DSIZE<1>         DSIZE<0>
;3062           RR.MM.P=5                        ;ROT RIGHT M'M, NO. BITS = PLATCH                       | SL.EQ.0          (PL<4:0>+SL).GT.32
;3063           RR.MM.PS=25                      ;ROT RIGHT M'M, NO. BITS = PLATCH+SLATCH                | SL.EQ.0          (PL<4:0>+SL).GT.32
;3064           RR.MM.SIZ=0E,   .VALIDITY=<V070> ;ROT RIGHT M'M, NO. BITS = 8,16,24,0                    | DSIZE<1>         DSIZE<0>
;3065           RR.RR.P=6                        ;ROT RIGHT R'R, NO. BITS = PLATCH                       | SL.EQ.0          (PL<4:0>+SL).GT.32
;3066           RR.RR.PS=26                      ;ROT RIGHT R'R, NO. BITS = PLATCH+SLATCH                | SL.EQ.0          (PL<4:0>+SL).GT.32
;3067           RR.RR.SIZ=0A,   .VALIDITY=<V070> ;ROT RIGHT R'R, NO. BITS = 8,16,24,0                    | DSIZE<1>         DSIZE<0>
;3068                                            ;                                                       |
;3069           ASL.R.P=28                       ;ARITH SHF LEFT  R, NO. BITS = PLATCH (NOTE 2)          | PL<4:0>.EQ.0     PL<5>
;3070           ASL.R.SIZ=17,   .VALIDITY=<V070> ;ARITH SHF LEFT  R, NO. BITS = 0,1,2,3                  | ASCII SIGN CHECK (SEE TABLE)
;3071           ASL.R.7=15                       ;ARITH SHF LEFT  R, NO. BITS = 7                        | ASCII SIGN CHECK (SEE TABLE)
;3072           ASL.M.P=29                       ;ARITH SHF LEFT  M, NO. BITS = PLATCH (NOTE 2)          | PL<4:0>.EQ.0     PL<5>
;3073                                            ;                                                       |
;3074           ASR.M.P=3                        ;ARITH SHF RIGHT M, NO. BITS = PLATCH                   |   0              PL<5>
;3075           ASR.M.-P=2A                      ;ARITH SHF RIGHT M, NO. BITS = -PLATCH                  | PL<4:0>.EQ.0     PL<5>
;3076           ASR.M.3=1D                       ;ARITH SHF RIGHT M, NO. BITS = 3                        | ASCII SIGN CHECK (SEE TABLE)
;3077                                            ;                                                       |
;3078           GETNIB=0F,      .VALIDITY=<V072> ;GET LEAST SIGNIFICANT NIBBLE FROM MBUS                 | DSIZE<1>         DSIZE<0>
;3079           BCDSWP=18                        ;BCD SWAP, MBUS                                         | S<3:0>.NE.0      S<3:0>.NE.(11,13)
;3080           CVTPN=1B                         ;CONVERT PACKED TO NUMERIC, 4NIB TO 4BYTE, MBUS         | S<3:0>.NE.0      S<3:0>.NE.(11,13)
;3081                                            ;RBUS MUST = 3XX33(HEX)                                 |
;3082           CVTNP=1F                         ;CONVERT NUMERIC TO PACKED, 8BYTE TO 8NIB, M'P          | ASCII SIGN CHECK (SEE TABLE)
;3083                                            ;                                                       |
;3084           PL_MSS=27                        ;FIND MOST SIGNIFICANT BIT SET MBUS,WBUS                | WX<31:16>.NE.0 WX<15:0>.NE.0
;3085           GETEXP=10                        ;EXTRACT & ZERO EXTEND M'M POS = 7, SIZE = 8            | S<3:0>.NE.0      S<3:0>.NE.(11,13)
;3086           GETFPF=19                        ;UNPACK FLOATING POINT FRACTION, M'R                    | S<3:0>.NE.0      S<3:0>.NE.(11,13)
;3087           FPLIT=1E                         ;EXPAND FLOATING POINT LITERAL, MBUS                    | ASCII SIGN CHECK (SEE TABLE)
;3088           FPACK=1A                         ;S ROT<31:16,15,14:7,6:0> <-                            | S<3:0>.NE.0      S<3:0>.NE.(11,13)
;3089                                            ; MB<24:9>,0,RB<7:0>,MB<31:25>                          |
```

```
;3090            PL=2C                          ;SUP ROT <- PLATCH                                  ! WBUS RANGE CHECK (SEE TABLE)
;3091            SL=2E                          ;SUP ROT <- SLATCH                                  ! WBUS RANGE CHECK (SEE TABLE)
;3092            SL.PL_WB=2F                    ;S ROT <- SLATCH . PLATCH  <- WB<5:0>               ! WBUS RANGE CHECK (SEE TABLE)
;3093            OLIT0.PL43_WB=3F               ;S ROT <- OLIT0 . PL<4:3> <- WB<1:0>                ! ABS VAL CHECK (SEE TABLE)
;3094            OLIT0.PL_LIT=3B                ;S ROT <- OLIT0 . PLATCH  <- SHORT LITERAL          ! ABS VAL CHECK (SEE TABLE)
;3095            PL.SL_WB=2D                    ;S ROT <- PLATCH . SLATCH  <- WB<5:0>               ! WBUS RANGE CHECK (SEE TABLE)
;3096            OLIT0.SL_LIT=3D                ;S ROT <- OLIT0 . SLATCH  <- SHORT LITERAL          ! ABS VAL CHECK (SEE TABLE)
;3097                                           ;
;3098            ZERO=16                        ;CONSTANT 0                                         ! ASCII SIGN CHECK (SEE TABLE)
;3099            MINUS1=39                      ;CONSTANT -1                                        ! ABS VAL CHECK (SEE TABLE)
;3100            CONX.SIZ=1C,   .VALIDITY=<V070>;CONSTANT 1,2,4,8 DEPENDING ON SIZE(-(R)+)          ! ASCII SIGN CHECK (SEE TABLE)
;3101            ZLIT0=30                       ;0 EXTEND LITERAL & ROT LEFT 00 BITS                ! ABS VAL CHECK (SEE TABLE)
;3102            ZLIT4=37                       ;0 EXTEND LITERAL & ROT LEFT 04 BITS                ! ABS VAL CHECK (SEE TABLE)
;3103            ZLIT8=36                       ;0 EXTEND LITERAL & ROT LEFT 08 BITS                ! ABS VAL CHECK (SEE TABLE)
;3104            ZLIT12=35                      ;0 EXTEND LITERAL & ROT LEFT 12 BITS                ! ABS VAL CHECK (SEE TABLE)
;3105            ZLIT16=34                      ;0 EXTEND LITERAL & ROT LEFT 16 BITS                ! ABS VAL CHECK (SEE TABLE)
;3106            ZLIT20=33                      ;0 EXTEND LITERAL & ROT LEFT 20 BITS                ! ABS VAL CHECK (SEE TABLE)
;3107            ZLIT24=32                      ;0 EXTEND LITERAL & ROT LEFT 24 BITS                ! ABS VAL CHECK (SEE TABLE)
;3108            ZLIT28=31                      ;0 EXTEND LITERAL & ROT LEFT 28 BITS                ! ABS VAL CHECK (SEE TABLE)
;3109            ZLITPL=2B                      ;0 EXTEND LITERAL & ROT LEFT PL BITS                ! PL<4:0>.EQ.0    0
;3110            OLIT0=38                       ;1 EXTEND LITERAL & ROT LEFT 00 BITS                ! ABS VAL CHECK (SEE TABLE)
;3111            OLIT8=3E                       ;1 EXTEND LITERAL & ROT LEFT 08 BITS                ! ABS VAL CHECK (SEE TABLE)
;3112            OLIT16=3C                      ;1 EXTEND LITERAL & ROT LEFT 16 BITS                ! ABS VAL CHECK (SEE TABLE)
;3113            OLIT24=3A                      ;1 EXTEND LITERAL & ROT LEFT 24 BITS                ! ABS VAL CHECK (SEE TABLE)
;3114
;3115
```

		SRKSTA<1>	SRKSTA<0>
ASCII SIGN CHECK	0=WB<7:0>.EQ.45 (ASCII -) 1=WB<7:0>.EQ.[32,43] (ASCII + OR SPACE) 2=MACHINE CHECK 3=WB<7:0>.NF.[32,43,45] (NOT = ASCII SIGN)	WB<7:0>.NE.(32,43,45)	WBUS<7:0>.NE.45
WBUS RANGE CHECK (UNSIGNED)	0=WB<7:0>.EQ.[1,31] 1=WB<7:0>.EQ.0 2=WB<7:0>.EQ.32 3=WB<7:0>.GT.32	WB<7:0>.GTU.31	WB<7:0>.NE.[1,32]
ABSOLUTE VALUE CHECK	0=WB<7:0>.EQ.[-1,-31] 1=WB<7:0>.LT.-31 2=WB<7:0>.EQ.[0,31] 3=WB<7:0>.GT.31	WB<7>.EQ.0	|WB<7:0>|.GE.32

NOTE 1 : When the rotate value is equal to zero, the super rotator will rotate by 32.
The results will be as expected for all other rotate values.

NOTE 2 : When the shift value is equal to zero, the super rotator will output zero.
The results will be as expected for all other shift values.

```
;3140   .TOC "  Machine Definition                : ROTSRK"
;3141
;3142   ROTSRK/=<63:58>                                ;SRKSTA<1>      SRKSTA<0>              XM      SHF      CI
;3143
;3144           ABSVAL.163.D=3B                       ;ABS VAL CHECK (SEE TABLE)             SIGN    ZERO     ZERO
;3145           ABSVAL.171.D=3D                       ;ABS VAL CHECK (SEE TABLE)             SIGN    ZERO     ZERO
;3146           ABSVAL.173.D=3F                       ;ABS VAL CHECK (SEE TABLE)             SIGN    ZERO     ZERO
;3147           ABSVAL.140=30                         ;ABS VAL CHECK (SEE TABLE)             SIGN    ALU0.Q1  ZERO
;3148           ABSVAL.141=31                         ;ABS VAL CHECK (SEE TABLE)             SIGN    ALU0.Q1  ALKC
;3149           ABSVAL.142=32                         ;ABS VAL CHECK (SEE TABLE)             SIGN    ALU0.Q1  ONE
;3150           ABSVAL.143=33                         ;ABS VAL CHECK (SEE TABLE)             SIGN    ALU0.Q1  PSLC
;3151           ABSVAL.150=34                         ;ABS VAL CHECK (SEE TABLE)             SIGN    ALU1.Q0  ZERO
;3152           ABSVAL.151=35                         ;ABS VAL CHECK (SEE TABLE)             SIGN    ALU1.Q0  ALKC
;3153           ABSVAL.152=36                         ;ABS VAL CHECK (SEE TABLE)             SIGN    ALU1.Q0  ONE
;3154           ABSVAL.153=37                         ;ABS VAL CHECK (SEE TABLE)             SIGN    ALU1.Q0  PSLC
;3155           ABSVAL.160=38                         ;ABS VAL CHECK (SEE TABLE)             SIGN    WBUS30   ZERO
;3156           ABSVAL.161=39                         ;ABS VAL CHECK (SEE TABLE)             SIGN    WBUS30   ALKC
;3157           ABSVAL.162=3A                         ;ABS VAL CHECK (SEE TABLE)             SIGN    WBUS30   ONE
;3158           ABSVAL.170=3C                         ;ABS VAL CHECK (SEE TABLE)             SIGN    PSLC     ZERO
;3159           ABSVAL.172=3E                         ;ABS VAL CHECK (SEE TABLE)             SIGN    PSLC     ONE
;3160
;3161           ASCIISIGN.050=14                      ;ASCII SIGN CHECK (SEE TABLE)          ZERO    ALU1.Q0  ZERO
;3162           ASCIISIGN.051=15                      ;ASCII SIGN CHECK (SEE TABLE)          ZERO    ALU1.Q0  ALKC
;3163           ASCIISIGN.052=16                      ;ASCII SIGN CHECK (SEE TABLE)          ZERO    ALU1.Q0  ONE
;3164           ASCIISIGN.053=17                      ;ASCII SIGN CHECK (SEE TABLE)          ZERO    ALU1.Q0  PSLC
;3165           ASCIISIGN.070=1C                      ;ASCII SIGN CHECK (SEE TABLE)          ZERO    PSLC     ZERO
;3166           ASCIISIGN.071=1D                      ;ASCII SIGN CHECK (SEE TABLE)          ZERO    PSLC     ALKC
;3167           ASCIISIGN.072=1E                      ;ASCII SIGN CHECK (SEE TABLE)          ZERO    PSLC     ONE
;3168           ASCIISIGN.073=1F                      ;ASCII SIGN CHECK (SEE TABLE)          ZERO    PSLC     PSLC
;3169
;3170           DSIZE.020=8,     .VALIDITY=<V072>     ;DSIZE<1>       DSIZE<0>               ZERO    SHF      ZERO
;3171           DSIZE.021=9,     .VALIDITY=<V072>     ;DSIZE<1>       DSIZE<0>               ZERO    SHF      ALKC
;3172           DSIZE.022=0A,    .VALIDITY=<V072>     ;DSIZE<1>       DSIZE<0>               ZERO    SHF      ONE
;3173           DSIZE.023=0B,    .VALIDITY=<V072>     ;DSIZE<1>       DSIZE<0>               ZERO    SHF      PSLC
;3174           DSIZE.030=0C,    .VALIDITY=<V072>     ;DSIZE<1>       DSIZE<0>               ZERO    ROT      ZERO
;3175           DSIZE.031=0D,    .VALIDITY=<V072>     ;DSIZE<1>       DSIZE<0>               ZERO    ROT      ALKC
;3176           DSIZE.032=0E,    .VALIDITY=<V072>     ;DSIZE<1>       DSIZE<0>               ZERO    ROT      ONE
;3177           DSIZE.033=0F,    .VALIDITY=<V072>     ;DSIZE<1>       DSIZE<0>               ZERO    SHF      ONE
;3178
;3179           PL.EQ.0.SIGN.120=28                   ;PL<4:0>.EQ.0   PL<5>                  SIGN    SHF      ZERO
;3180           PL.EQ.0.SIGN.121=29                   ;PL<4:0>.EQ.0   PL<5>                  SIGN    SHF      ALKC
;3181           PL.EQ.0.SIGN.122=2A                   ;PL<4:0>.EQ.0   PL<5>                  SIGN    SHF      ONE
;3182           PL.EQ.0.123=2B                        ;PL<4:0>.EQ.0   0                      SIGN    SHF      PSLC
;3183
;3184           BCDSIGN.040=10                        ;S<3:0>.NE.0    S<3:0>.NE.(11,13)      ZERO    ALU0.Q1  ZERO
;3185           BCDSIGN.041=11                        ;S<3:0>.NE.0    S<3:0>.NE.(11,13)      ZERO    ALU0.Q1  ALKC
;3186           BCDSIGN.042=12                        ;S<3:0>.NE.0    S<3:0>.NE.(11,13)      ZERO    ALU0.Q1  ONE
;3187           BCDSIGN.043=13                        ;S<3:0>.NE.0    S<3:0>.NE.(11,13)      ZERO    ALU0.Q1  PSLC
;3188           BCDSIGN.060=18                        ;S<3:0>.NE.0    S<3:0>.NE.(11,13)      ZERO    WBUS30   ZERO
;3189           BCDSIGN.061=19                        ;S<3:0>.NE.0    S<3:0>.NE.(11,13)      ZERO    WBUS30   ALKC
;3190           BCDSIGN.062=1A                        ;S<3:0>.NE.0    S<3:0>.NE.(11,13)      ZERO    WBUS30   ONE
;3191           BCDSIGN.063=1B                        ;S<3:0>.NE.0    S<3:0>.NE.(11,13)      ZERO    WBUS30   PSLC
```

```
;3192          VIELD.000=0                        ;SL.EQ.0        (PL<4:0>+SL).GT.32      ZERO    ZERO    ZERO
;3193          VIELD.001=1                        ;SL.EQ.0        (PL<4:0>+SL).GT.32      ZERO    ZERO    ALKC
;3194          VIELD.002=2                        ;SL.EQ.0        (PL<4:0>+SL).GT.32      ZERO    ZERO    ONE
;3195          VIELD.010=4                        ;SL.EQ.0        (PL<4:0>+SL).GT.32      ZERO    ONE     ZERO
;3196          VIELD.011=5                        ;SL.EQ.0        (PL<4:0>+SL).GT.32      ZERO    ONE     ALKC
;3197          VIELD.012=6                        ;SL.EQ.0        (PL<4:0>+SL).GT.32      ZERO    ONE     ONE
;3198          VIELD.110=24                       ;SL.EQ.0        (PL<4:0>+SL).GT.32      SIGN    ONE     ZERO
;3199          VIELD.111=25                       ;SL.EQ.0        (PL<4:0>+SL).GT.32      SIGN    ONE     ALKC
;3200          VIELD.112=26                       ;SL.EQ.0        (PL<4:0>+SL).GT.32      SIGN    ONE     ONE
;3201
;3202          SL.EQ.0.SIGN.101=21                ;SL.EQ.0        PL<5>                   SIGN    ZERO    ALKC
;3203          SL.EQ.0.SIGN.102=22                ;SL.EQ.0        PL<5>                   SIGN    ZERO    ONE
;3204          SL.EQ.0.100=20                     ;SL.EQ.0        UNDEFINED               SIGN    ZERO    ZERO
;3205
;3206          WBRANGE.131.D=2D                   ;WBUS RANGE CHECK (SEE TABLE)           SIGN    ZERO    ZERO
;3207          WBRANGE.133.D=2F                   ;WBUS RANGE CHECK (SEE TABLE)           SIGN    ZERO    ZERO
;3208          WBRANGE.130=2C                     ;WBUS RANGE CHECK (SEE TABLE)           SIGN    ROT     ZERO
;3209          WBRANGE.132=2E                     ;WBUS RANGE CHECK (SEE TABLE)           SIGN    ROT     ONE
;3210
;3211          WX.NE.0.113.D=27                   ;WX<31:16>.NE.0 WX<15:0>.NE.0           SIGN    ZERO    ZERO
;3212          WX.NE.0.103=23                     ;WX<31:16>.NE.0 WX<15:0>.NE.0           SIGN    ZERO    PSLC
;3213
;3214          PL5.003=3                          ;0              PL<5>                   ZERO    ZERO    PSLC
;3215          PL5.013=7                          ;0              PL<5>                   ZERO    ONE     PSLC
```

```
;3216    .TOC "  Machine Definition                : RSRC, SPW"
;3217
;3218    RSRC/=<39:34>,.DEFAULT=0;R SCRATCH PAD ADD CTRL RTMP=0-15,GPR=16-31,IPR=32-47
;3219                            ;DEFAULT TO RTMP7 FOR LIT/(LITRL,LONLIT)
;3220
;3221            TEMP0=00        ;RTMP - DUAL PORT TEMP 0
;3222            TEMP1=01        ;RTMP - DUAL PORT TEMP 1
;3223            TEMP2=02        ;RTMP - DUAL PORT TEMP 2
;3224            TEMP3=03        ;RTMP - DUAL PORT TEMP 3
;3225            TEMP4=04        ;RTMP - DUAL PORT TEMP 4
;3226            TEMP5=05        ;RTMP - DUAL PORT TEMP 5
;3227            TEMP6=06        ;RTMP - DUAL PORT TEMP 6
;3228            TEMP7=07        ;RTMP - DUAL PORT TEMP 7
;3229            TEMP8=08        ;RTMP - R S-PAD TEMP 8
;3230            TEMP9=09        ;RTMP - R S-PAD TEMP 9
;3231            TEMP10=0A       ;RTMP - R S-PAD TEMP 10
;3232            TEMP11=0B       ;RTMP - R S-PAD TEMP 11
;3233            TEMP12=0C       ;RTMP - R S-PAD TEMP 12
;3234            TEMP13=0D       ;RTMP - R S-PAD TEMP 13
;3235            MM.TEMP5=0E     ;MEMORY MANAGEMENT TEMP 5
;3236            MM.TEMP1=0F     ;MEMORY MANAGEMENT TEMP 1
;3237
;3238            R0=10           ;GPR  - GENERAL PURPOSE REGISTER 0
;3239            R1=11           ;GPR  - GENERAL PURPOSE REGISTER 1
;3240            R2=12           ;GPR  - GENERAL PURPOSE REGISTER 2
;3241            R3=13           ;GPR  - GENERAL PURPOSE REGISTER 3
;3242            R4=14           ;GPR  - GENERAL PURPOSE REGISTER 4
;3243            R5=15           ;GPR  - GENERAL PURPOSE REGISTER 5
;3244            R6=16           ;GPR  - GENERAL PURPOSE REGISTER 6
;3245            R7=17           ;GPR  - GENERAL PURPOSE REGISTER 7
;3246            R8=18           ;GPR  - GENERAL PURPOSE REGISTER 8
;3247            R9=19           ;GPR  - GENERAL PURPOSE REGISTER 9
;3248            R10=1A          ;GPR  - GENERAL PURPOSE REGISTER 10
;3249            R11=1B          ;GPR  - GENERAL PURPOSE REGISTER 11
;3250            R12=1C          ;GPR  - GENERAL PURPOSE REGISTER 12
;3251            R13=1D          ;GPR  - GENERAL PURPOSE REGISTER 13
;3252            SP=1E           ;GPR  - STACK POINTER
;3253            RTMPGPR=1F      ;GPR  - MICRO CODE TEMPORARY
;3254
;3255            KSP=20          ;IPR  - KERNEL STACT POINTER
;3256            ESP=21          ;IPR  - EXECUTIVE STACK POINTER
;3257            SSP=22          ;IPR  - SUPERVISOR STACT POINTER
;3258            USP=23          ;IPR  - USER STACK POINTER
;3259            ISP=24          ;IPR  - INTERRUPT STACK POINTER
;3260            PCBB=25         ;IPR  - PROCESS CONTROL BLOCK BASE
;3261            MM.TEMP2=26     ;MEMORY MANAGEMENT TEMP 2
;3262            MM.TEMP3=27     ;MEMORY MANAGEMENT TEMP 3
;3263            P0BR=28         ;IPR  - P0 BASE REGISTER
;3264            P0LR=29         ;IPR  - P0 LENGTH REGISTER
;3265            P1BR=2A         ;IPR  - P1 BASE REGISTER
;3266            P1LR=2B         ;IPR  - P1 LENGTH REGISTER
;3267            SBR=2C          ;IPR  - SYSTEM BASE REGISTER
;3268            SLR=2D          ;IPR  - SYSTEM LENGTH REGISTER
;3269            SPNICR.SPICR=2E ;IPR  - NEXT INTERVAL REGISTER
;3270            MM.TEMP4=2F     ;MEMORY MANAGEMENT TEMP 4
```

```
;3271            TEMP.R=30        ;RTMP INDEXED BY RNUM
;3272            TEMP.R+1=3C      ;RTMP INDEXED BY RNUM+1
;3273            TEMP.ROR1=38     ;RTMP INDEXED BY RNUM.OR.1
;3274            GPR.R=33         ;GPR  INDEXED BY RNUM
;3275            GPR.R+1=3F       ;GPR  INDEXED BY RNUM+1
;3276            GPR.ROR1=3B      ;GPR  INDEXED BY RNUM.OR.1
;3277            DST.R=31         ;GPR  INDEXED BY RNUM          INH WRT IF NOT REG MODE
;3278            DST.R+1=3D       ;GPR  INDEXED BY RNUM+1        INH WRT IF NOT REG MODE
;3279            DST.ROR1=39      ;GPR  INDEXED BY RNUM.OR.1     INH WRT IF NOT REG MODE
;3280            IPR.R=32         ;IPR  INDEXED BY RNUM
;3281            IPR.R+1=3E       ;IPR  INDEXED BY RNUM+1
;3282            IPR.ROR1=3A      ;IPR  INDEXED BY RNUM.OR.1
;3283
;3284            LONLIT=35        ;RBUS <- LONLIT REGISTER       DEFAULT FOR WRT = RTMP0
;3285            ZERO=36          ;RBUS <- 0                     DEFAULT FOR WRT = RTMP0
;3286            ZERO.CLRRBSP=37  ;RBUS <- 0  .  RBSP <- 0       DEFAULT FOR WRT = RTMP0
;3287
;3288    SPW/=<70:69>,.DEFAULT=0 ;SCRATCH PAD WRITE CONTROL
;3289            NOP=0                                  ;NO-OPERATION
;3290            RSIZE=1,         .VALIDITY=<V070>      ;WRITE SCRATCH PAD R SIZE DEPENDENT
;3291            RLONG=2                                ;WRITE SCRATCH PAD R SIZE IS LONG
;3292            MLONG=3                                ;WRITE SCRATCH PAD M SIZE IS LONG
```

```
;3293   .TOC "  Machine Definition                : WCTRL"
;3294
;3295   WCTRL/=<30:25>,.DEFAULT=2                          ;WBUS CONTROL
;3296
;3297                                                      ;MISC
;3298           NOP=2                                              ;NO OPERATION
;3299           ACLOSYNC=1D,              .VALIDITY=<V080>         ;WBUS<16> <- ACLO SYNC
;3300           FPA.ENABLE_WB5=15                                  ;FPA ENABLE <- WBUS<5>
;3301           REVLEVEL=1T,              .VALIDITY=<V080>         ;WBUS<23:16> <- REVISION LEVEL
;3302
;3303                                                      ;INTERRUPT AND EXCEPTIONS
;3304           FPTCR=03,                 .VALIDITY=<V080>         ;WBUS<2:0> <- FLOATING POINT TRAP CODE REGISTER
;3305           ASTLVL=3A,                .VALIDITY=<V00.30.80>    ;WBUS<26:24> <- ASTLVL
;3306           ASTLVL_WB=38,             .VALIDITY=<V00.30>       ;ASTLVL <- WBUS<26:24>
;3307           SOFTIPR_WB=3C,            .VALIDITY=<V00.30>       ;SOFTWARE IPR REGISTER <- WBUS<20:16>
;3308           PREV_CUR.ISCUR_WB=35,     .VALIDITY=<V01.30>       ;PSL<PREV> <- PSL<CUR> . PSL<IS,CUR> <- WB<26:24>
;3309           PREV_WB=31,               .VALIDITY=<V00.30>       ;PSL<PREV> <- WBUS<23:22>
;3310           IPL_WB=3D,                .VALIDITY=<V00.30>       ;PSL<IPL> <- WBUS<20:16>
;3311           IPR=3F,                   .VALIDITY=<V00.30.80>    ;WBUS<20:16> <- IPL OF HIGHEST IPR
;3312           UVCTR_CM.IS=10                                     ;UVCTR<3:0> <- UNP'IS'CM<1:0>
;3313           REICHR=37,                .VALIDITY=<V05.30>       ;REI CHECK
;3314           GRANT=33,                 .VALIDITY=<V06.36.40>    ;ISSUE UNIBUS GRANT
;3315
;3316                                                      ;MICRO SEQUENCER FUNCTIONS
;3317           STEPC_WB=8                                         ;STEP COUNTER <- WBUS<4:0>
;3318           CM.TP.FPD.F5.STEPC=0C,    .VALIDITY=<V080>         ;WBUS<31:30,27,5:0> <- PSL<CM,TP,FPD>, FLAG5, STEP COUNTER
;3319           FLAGS_WB=18                                        ;STATUS FLAGS <- WBUS<5:0>
;3320           CM.TP.FPD.FLAGS=1C,       .VALIDITY=<V080>         ;WBUS<31:30,27,5:0> <- PSL<CM.TP.FPD>, STATUS FLAGS
;3321
;3322                                                      ;TIMER CONTROL & TOD CLOCK
;3323           TCSR_WB=0B                                         ;TIMER CONTROL AND STATUS REG. <- WBUS<31,24:16>
;3324           INIR=0E,                  .VALIDITY=<V080>         ;WBUS<15:0> <- INTERNAL NEXT INTERVAL REG.
;3325           TCSR.IICR=0F,             .VALIDITY=<V080>         ;WBUS<31,24:16> <- TCSR . WBUS<15:0> <- IICR
;3326           INIR_WB=0A                                         ;INTERNAL NEXT INTERVAL REG <- WBUS<15:0>
;3327           TODCLK_WB=0D                                       ;TOD CLOCK <- WBUS<31:0>
;3328           TODCLK=9,                 .VALIDITY=<V080>         ;WBUS<31:0> <- TOD CLOCK
;3329
;3330                                                      ;CONSOLE & TU58 INTERFACE CONTROL
;3331           LOADCRAR=12                                        ;CRAR <- WBUS<23:22>                    (SEE TABLE 2)
;3332           CONWRITE=16                                        ;WRITE CONSOLE REGISTER PER CRAR        (SEE TABLE 2)
;3333           CONREAD=1E,               .VALIDITY=<V080>         ;READ  CONSOLE REGISTER PER CRAR        (SEE TABLE 2)
;3334           READCRAR=1A,              .VALIDITY=<V080>         ;WBUS<31:30> <- CRAR                    (SEE TABLE 2)
;3335           LOADTRAR=13                                        ;TRAR <- WBUS<23:22>                    (SEE TABLE 3)
;3336           TU58WRITE=17                                       ;WRITE TU58 REGISTER PER TRAR           (SEE TABLE 3)
;3337           TU58READ=1F,              .VALIDITY=<V080>         ;READ  TU58 REGISTER PER TRAR           (SEE TABLE 3)
;3338           READTRAR=1B,              .VALIDITY=<V080>         ;WBUS<31:30> <- TRAR                    (SEE TABLE 3)
;3339
;3340                                                      ;ADDRESS CONTROL
;3341           PC_WB=24,                 .VALIDITY=<V00.30.41>    ;PC <- WBUS
;3342           PC_PC+WB=2C,              .VALIDITY=<V00.30.41>    ;PC <- PC + WBUS
;3343           VA_WB=25,                 .VALIDITY=<V00.30>       ;VA <- WBUS
;3344           VA_VA+4=22,               .VALIDITY=<V00.30>       ;VA <- VA + 4
;3345           VA_PC+I+W.PC_PC+I=20,     .VALIDITY=<V01.31.40.70> ;VA <- PC+ISIZE+WBUS . PC <- PC+ISIZE
;3346                                                              ;(NO BUS CYCLE OR BUT XB DURING THIS CYCLE)
```

```
;3347                                                                   ;MEMORY CONTROL
;3348            PME=14                                                 ;PERFORMANCE MONITOR ENABLE
;3349            MDR_WB=23,              .VALIDITY=<V02.33>             ;MDR <- WBUS
;3350            MDR_0=27,               .VALIDITY=<V02.21>             ;MDR <- 0
;3351            MDR_IR=2B,              .VALIDITY=<V02.33.41>          ;MDR <- IR ZERO-EXTENDED
;3352            WDR_WB=2E,              .VALIDITY=<V02.33>             ;WDR <- WBUS ROTATED FOR LONGWORD ALIGNMENT
;3353            WDR_WB.UR=2A,           .VALIDITY=<V02.33>             ;WDR <- WBUS UN-ROTATED
;3354            MBUS_WDR=26,            .VALIDITY=<V07.37>             ;MBUS <- WDR                (NOTE: MSRC=WDR)
;3355            TB_WB=28,               .VALIDITY=<V11.34>             ;TRANSLATION BUFFER DATA <- WBUS
;3356            CLRTB.VA_WB=29,         .VALIDITY=<V10.35>             ;TB VALID BIT <- 0      (INVALIDATE BOTH GROUPS AT THE
;3357                                                                   ;VA <- WBUS              INDEX POSITION ADDRESSED BY VA)
;3358            CLRCH.VA_WB=2D,         .VALIDITY=<V04.33>             ;CACHE VALID BIT <- 0   (INVALIDATE BOTH GROUPS AT THE
;3359                                                                   ;VA <- WBUS              INDEX POSITION ADDRESSED BY VA)
;3360            MEMSCAR_WB=34,          .VALIDITY=<V00.30>             ;MEMORY STATUS & CONTROL ADDRESS REG <- WBUS<27:24>
;3361            MEMSCR=32,              .VALIDITY=<V01.32.40.80>       ;WBUS<27:24> <- MEMORY STATUS & CONTROL REG (@MSCAR)
;3362            MEMSCR_WB=30,           .VALIDITY=<V01.32.40>          ;MEMORY STATUS & CONTROL REG(@MEMSCAR) <- WBUS<27:24>
;3363
;3364    .CREF                           ;END OF CREF ONLY WHEN FULL ASSEMBLY THE POLLOWING WILL ALWAYS CREF
;3365
;3366
;3367    .NOCREF                 ;SET UP FOR NEVER CREF
;3368    .NOCREF
;3369
;3370    PAR1/=<78:78>,.DEFAULT=<,NOTE,PARITY[<CC/>,<RSRC/>,<DTYPE/>,<BUT/>,<APCTL/>,<ROT/>,<MSRC/>,<SPW/>]]>          ;EVEN PARITY
;3371    PAR2/=<79:79>,.DEFAULT=<      ,PARITY[<NEXT/>,<JSR/>,<CLKX/>,<FPA/>,<BUS/>,<WCTRL/>,<ISTRM/>,<MISC/>,<LIT/>]>   ;ODD PARITY
;3372
;3373
;3374    VSIZE/=<84:84>,.DEFAULT=0                                      ;DSIZE CHECK
```

```
;3375    .TOC "  Machine Definition              : IRD1 ROM"
;3376    .ICODE
;3377    .WIDTH/32
;3378
;3379
;3380
;3381    ; +---+---+---+---------------+---+---------------+---+---------------+---+---------------+
;3382    ; | V | V | I |               | I |               | F |               | F |               |
;3383    ; | I | F | F |   IRD1.FPA    | O |     IRD1      | F |   FPD.FPA     | O |     FPD       |
;3384    ; | R | P | O |               | P |               | O |               | P |               |
;3385    ; | D | D | P |               |   |               | P |               |   |               |
;3386    ; | 1 |   |   |               |   |               |   |               |   |               |
;3387    ; +---+---+---+---------------+---+---------------+---+---------------+---+---------------+
;3388    ; | 3 | 3 | 3 | 3 2 2 2 2 2 2 2 | 2 | 2 2 2 1 1 1 1 1 | 1 1 1 1 1 0 0 0 | 0 0 0 0 0 0 0 0 |
;3389    ; | 3 | 2 | 1 | 0 9 8 7 6 5 4 3 | 3 | 2 1 0 9 8 7 6 5 | 5 4 3 2 1 0 9 8 | 7 6 5 4 3 2 1 0 |
;3390    ; +---+---+---+---------------+---+---------------+---+---------------+---+---------------+
;3391
;3392    FPD      /=<6:0>
;3393    FPD.FPA  /=<14:8>
;3394    IRD1     /=<22:16>
;3395    IRD1.FPA/=<30:24>
;3396
;3397    FOP  /=<07:07>
;3398             NOP=0
;3399             LOD=1
;3400    FFOP/=<15:15>
;3401             NOP=0
;3402             LOD=1
;3403    IOP  /=<23:23>
;3404             NOP=0
;3405             LOD=1
;3406    IFOP/=<31:31>
;3407             NOP=0
;3408             LOD=1
;3409
;3410    VFPD /=<32:32>,          .VALIDITY=<V060>
;3411    VIRD1/=<33:33>,          .VALIDITY=<V061>
```

```
;3412    .TOC "  Machine Definition                : IRDX ROM"
;3413    .OCODE
;3414    .WIDTH/96
;3415
;3416
;3417
;3418
;3419
;3420
;3421
;3422
;3423
;3424
;3425
;3426
;3427
;3428
;3429
;3430
;3431
;3432
;3433
;3434
;3435
;3436
;3437
;3438
;3439    CNT0.FPA.MEM/=<10:0>
;3440    CNT0.FPA.REG/=<21:11>,   .VALIDITY=<V062>
;3441    0FOP/=<23:22>
;3442            NOP=0
;3443            LOD=3
;3444
;3445    CNT0.MEM/=<34:24>
;3446    CNT0.REG/=<45:35>,       .VALIDITY=<V063>
;3447    00P /=<47:46>
;3448            NOP=0
;3449            LOD=3
;3450
;3451    CNT1.FPA.MEM/=<58:48>
;3452    CNT1.FPA.REG/=<69:59>,   .VALIDITY=<V064>
;3453    1FOP/=<71:70>
;3454            NOP=0
;3455            LOD=3
;3456
;3457    CNT1.MEM/=<82:72>
;3458    CNT1.REG/=<93:83>,       .VALIDITY=<V065>
;3459    10P /=<95:94>
;3460            NOP=0
;3461            LOD=3
;3462
;3463    VCNT0/=<96:96>,          .VALIDITY=<V066>
;3464    VCNT1/=<97:97>,          .VALIDITY=<V067>
```

0 0 P	CNT0.REG	CNT0.MEM	0 F O P	CNT0.FPA.REG	CNT0.FPA.MEM
47 46	45 44 43 42 41 40 39 38 37 36 35	34 33 32 31 30 29 28 27 26 25 24	23 22	21 20 19 18 17 16 15 14 13 12 11	10 9 8 7 6 5 4 3 2 1 0

V C N T 1	V C N T 0	1 0 P	CNT1.REG	CNT1.MEM	1 F O P	CNT1.FPA.REG	CNT1.FPA.MEM
97	96	95 94	93 92 91 90 89 88 87 86 85 83	82 81 80 79 78 77 76 75 74 73 72	71 70	69 68 67 66 65 64 63 62 61 60 59	58 57 56 55 54 53 52 51 50 49 48

```
;3465   .TOC "  Machine Definition                : COMPATABILITY ROM"
;3466   .CCODE
;3467   .WIDTH/66
;3468
;3469   ;       +---------------------+---------------------+---------------------+
;3470   ;       |                     |                     |                     |
;3471   ;       |      CNT1.MEM       |      CNT0.MEM       |      IRD1.MEM       |
;3472   ;       |                     |                     |                     |
;3473   ;       +---------------------+---------------------+---------------------+
;3474   ;       |3 3 3 2 2 2 2 2 2 2 2|2 2 1 1 1 1 1 1 1 1 1|1 0 0 0 0 0 0 0 0 0 0|
;3475   ;       |2 1 0 9 8 7 6 5 4 3 2|1 0 9 8 7 6 5 4 3 2 1|0 9 8 7 6 5 4 3 2 1 0|
;3476   ;       +---------------------+---------------------+---------------------+
;3477   ;
;3478   ;
;3479   ;  +-+-+---------------------+---------------------+---------------------+
;3480   ;  |V|V|                     |                     |                     |
;3481   ;  |R|M|      CNT1.REG       |      CNT0.REG       |      IRD1.REG       |
;3482   ;  |F|E|                     |                     |                     |
;3483   ;  |G|M|                     |                     |                     |
;3484   ;  +-+-+---------------------+---------------------+---------------------+
;3485   ;  |6|6|6 6 6 6 6 6 5 5 5 5 5|5 5 5 5 5 4 4 4 4 4 4|4 4 4 4 3 3 3 3 3 3 3|
;3486   ;  |7|6|5 4 3 2 1 0 9 8 7 6 5|4 3 2 1 0 9 8 7 6 5 4|3 2 1 0 9 8 7 6 5 4 3|
;3487   ;  +-+-+---------------------+---------------------+---------------------+
;3488
;3489   IRD1.MEM/=<10:00>
;3490   CNT0.MEM/=<21:11>
;3491   CNT1.MEM/=<32:22>
;3492
;3493   IRD1.REG/=<43:33>
;3494   CNT0.REG/=<54:44>
;3495   CNT1.REG/=<65:55>
;3496
;3497   VMEM/=<66:66>,              .VALIDITY=<V068>
;3498   VREG/=<67:67>,              .VALIDITY=<V069>
```

```
;3499    .TOC "  Machine Definition              : DSIZE ROM"
;3500    .DCODE
;3501    ;  +---+---+---+---+---+---+
;3502    ;  | 0 | 0 | 0 | 0 | 0 | 0 |
;3503    ;  | S | S | S | S | S | S |
;3504    ;  | 6 | 5 | 4 | 3 | 2 | 1 |
;3505    ;  +---+---+---+---+---+---+
;3506    ;  |1 1|0 0|0 0|0 0|0 0|0 0|
;3507    ;  |1 0|9 8|7 6|5 4|3 2|1 0|
;3508    ;  +---+---+---+---+---+---+
;3509    OS1/=<1:0>
;3510            BYTE=0
;3511            WORD=1
;3512            LONG=2
;3513            FLOT=2
;3514            QUAD=3
;3515            DBLE=3
;3516    OS2/=<3:2>
;3517            BYTE=0
;3518            WORD=1
;3519            LONG=2
;3520            FLOT=2
;3521            QUAD=3
;3522            DBLE=3
;3523    OS3/=<5:4>
;3524            BYTE=0
;3525            WORD=1
;3526            LONG=2
;3527            FLOT=2
;3528            QUAD=3
;3529            DBLE=3
;3530    OS4/=<7:6>
;3531            BYTE=0
;3532            WORD=1
;3533            LONG=2
;3534            FLOT=2
;3535            QUAD=3
;3536            DBLE=3
;3537    OS5/=<9:8>
;3538            BYTE=0
;3539            WORD=1
;3540            LONG=2
;3541            FLOT=2
;3542            QUAD=3
;3543            DBLE=3
;3544    OS6/=<11:10>
;3545            BYTE=0
;3546            WORD=1
;3547            LONG=2
;3548            FLOT=2
;3549            QUAD=3
;3550            DBLE=3
;3551    .UCODE
;3552    .CREF
;3553    .CREF                   ;END OF NEVER CREF THE FOLLOWING WILL ALWAYS CREF
```

```
;3554    .TOC     "       Validity Checks                  : Combinations"
;3555
;3556    .NOCREF                    ;SET UP FOR CREF ONLY WHEN FULL ASSEMBLY
;3557
;3558             .SET/V00.30        =<.AND[<V000>,<V030>]>
;3559             .SET/V00.30.80     =<.AND[<V000>,<V030>,<V080>]>
;3560             .SET/V00.30.41     =<.AND[<V000>,<V030>,<V041>]>
;3561             .SET/V00.71        =<.AND[<V000>,<V071>]>
;3562             .SET/V00.71.80     =<.AND[<V000>,<V071>,<V080>]>
;3563             .SET/V01.30        =<.AND[<V001>,<V030>]>
;3564             .SET/V01.31.40     =<.AND[<V001>,<V031>,<V040>]>
;3565             .SET/V01.31.40.70=<.AND[<V001>,<V031>,<V040>,<V070>]>
;3566             .SET/V01.32.40     =<.AND[<V001>,<V032>,<V040>]>
;3567             .SET/V01.32.40.80=<.AND[<V001>,<V032>,<V040>,<V080>]>
;3568             .SET/V02.21        =<.AND[<V002>,<V021>]>
;3569             .SET/V02.33        =<.AND[<V002>,<V033>]>
;3570             .SET/V02.33.41     =<.AND[<V002>,<V033>,<V041>]>
;3571             .SET/V04.33        =<.AND[<V004>,<V033>]>
;3572             .SET/V04.34        =<.AND[<V004>,<V034>]>
;3573             .SET/V04.35        =<.AND[<V004>,<V035>]>
;3574             .SET/V05.30        =<.AND[<V005>,<V030>]>
;3575             .SET/V06.36.40     =<.AND[<V006>,<V036>,<V040>]>
;3576             .SET/V07.37        =<.AND[<V007>,<V037>]>
;3577             .SET/V09.21        =<.AND[<V009>,<V021>]>
;3578             .SET/V10.35        =<.AND[<V010>,<V035>]>
;3579             .SET/V11.34        =<.AND[<V011>,<V034>]>
;3580             .SET/V20.70        =<.AND[<V020>,<V070>]>
;3581             .SET/V22.70        =<.AND[<V022>,<V070>]>
;3582             .SET/V24.25.70     =<.AND[<V024>,<V025>,<V070>]>
;3583             .SET/V21.50-54.70=<.AND[<V021>,<V050>,<V051>,<V052>,<V053>,<V054>,<V070>]>
;3584             .SET/V50.51.70     =<.AND[<V050>,<V051>,<V070>]>
;3585             .SET/V71.80        =<.AND[<V071>,<V080>]>
```

```
;3586    .TOC    ''      Validity Checks                 : WCTRL/CCPSL vs BUS Conflict States''
;3587            .SET/V000=<.CASE[<BUS/>]OF[1,0,1,0,1,1,1,1,1,0,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1]>
;3588            .SET/V001=<.CASE[<BUS/>]OF[0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,1,1,0,0,0,0,1,1,1,1]>
;3589            .SET/V002=<.CASE[<BUS/>]OF[0,0,0,0,0,1,0,1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1]>
;3590            .SET/V004=<.CASE[<BUS/>]OF[0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]>
;3591            .SET/V005=<.CASE[<BUS/>]OF[0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]>
;3592            .SET/V006=<.CASE[<BUS/>]OF[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]>
;3593            .SET/V007=<.CASE[<BUS/>]OF[0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1]>
;3594            .SET/V008=<.EQL[<WCTRL/>,<WCTRL/REICHK>]>
;3595            .SET/V009=<.EQL[<WCTRL/>,<WCTRL/GRANT>]>
;3596            .SET/V010=<.ORC   <.AND[  <.EQL[<BUS/>,<BUS/PRB.RD.PTE>]>   ,  <.NEQ[<BUT/>,<BUT/UVCTR>]>  ]>,
;3597                              <.CASE[<BUS/>]OF[0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]>   ]>
;3598            .SET/V011=<.ORC   <.AND[  <.EQL[<BUS/>,<BUS/PRB.RD>]>   ,  <.NEQ[<BUT/>,<BUT/UVCTR>]>  ]>,
;3599                              <.CASE[<BUS/>]OF[0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]>   ]>
;3600
;3601    .TOC    ''      Validity Checks                 : BUS vs MSRC Conflict States''
;3602            .SET/V020=<.CASE[<MSRC/>]OF[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,<INIT>,0,0,0,0,1,1,1,0]>
;3603            .SET/V021=<.CASE[<MSRC/>]OF[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,    0,0,0,0,0,1,1,1,0]>
;3604            .SET/V022=<.CASE[<MSRC/>]OF[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,<INIT>,0,0,0,0,1,1,1,0]>
;3605            .SET/V023=<.CASE[<MSRC/>]OF[1,1,1,1,0,0,0,0,1,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,    0,0,0,0,0,0,0,0,0]>
;3606            .SET/V024=<.ORC   <.NEQ[<BUT/>,<BUT/UVCTR>]>,
;3607                              <.CASE[<MSRC/>]OF[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,1,1,1,0]>   ]>
;3608            .SET/V025=<.CASE[<MSRC/>]OF[1,1,1,1,0,0,0,0,1,1,1,1,1,0,0,0,0,1,1,1,1,1,0,0,0,    0,1,1,1,1,1,0,0,0,1]>
;3609
;3610    .TOC    ''      Validity Checks                 : WCTRL/CCPSL vs MSRC Conflict States''
;3611            .SET/V030=<.CASE[<MSRC/>]OF[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1]>
;3612            .SET/V031=<.CASE[<MSRC/>]OF[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0]>
;3613            .SET/V032=<.CASE[<MSRC/>]OF[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1]>
;3614            .SET/V033=<.CASE[<MSRC/>]OF[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,0]>
;3615            .SET/V034=<.CASE[<MSRC/>]OF[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0]>
;3616            .SET/V035=<.CASE[<MSRC/>]OF[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,0,0,1,1,1,1,1]>
;3617            .SET/V036=<.CASE[<MSRC/>]OF[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,0,1,1,1,1,1,1,0]>
;3618            .SET/V037=<.CASE[<MSRC/>]OF[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0]>
;3619            .SET/V038=<.EQL[<WCTRL/>,<WCTRL/MBUS_WDR>]>
;3620
;3621    .TOC    ''      Validity Checks                 : Buts That Take Data From XB''
;3622            .SET/V040=<.NEQ[<BUT/>,<BUT/IRD1>,<BUT/IRD1TST>,<BUT/LOD.INC.BRA>,<BUT/LOD.BRA>]>
;3623            .SET/V041=<.NEQ[<BUT/>,<BUT/IRD1>,<BUT/IRD1TST>]>
;3624
;3625    .TOC    ''      Validity Checks                 : Special ALPCTL Functions For Multiply and Divide''
;3626            .SET/V050=<.EQL[<ROT/>,0]>
;3627            .SET/V051=<.EQL[<BUS/>,<BUS/NOP>]>
;3628            .SET/V052=<.NEQ[<WCTRL/>,<WCTRL/TB_WB>,<WCTRL/CLRTB.VA_WB>,<WCTRL/CLRCH.VA_WB>,<WCTRL/MDR_WB>,<WCTRL/MDR_0>,
;3629                                     <WCTRL/MDR_IR>,<WCTRL/WDR_WB>,<WCTRL/WDR_WB.UR>,<WCTRL/MBUS_WDR>,<WCTRL/GRANT>,
;3630                                     <WCTRL/REICHK>]>
;3631            .SET/V053=<.NEQ[<CCPSL/>,<CCPSL/MDR_OSR.CCBR_BRATST>]>
;3632            .SET/V054=<.NEQ[<BUT/>,<BUT/IRDX>,<BUT/IRD1>,<BUT/IRD1TST>,<BUT/LOD.INC.BRA>,<BUT/LOD.BRA>]>
```

```
;3633   .TOC    "       Validity Checks                 : IRD Rom Checks"
;3634           .SET/V060=<VFPD/>
;3635           .SET/V061=<VIRD1/>
;3636           .SET/V062=<.OR[   <.EQL[<OFOP/>,<OFOP/LOD>]>    ,   <.EQL[<CNT0.FPA.MEM/>,<CNT0.FPA.REG/>]>   ]>
;3637           .SET/V063=<.OR[   <.EQL[<OOP/> ,<OOP/LOD>]>     ,   <.EQL[<CNT0.MEM/>     ,<CNT0.REG/>]>      ]>
;3638           .SET/V064=<.OR[   <.EQL[<1FOP/>,<1FOP/LOD>]>    ,   <.EQL[<CNT1.FPA.MEM/>,<CNT1.FPA.REG/>]>   ]>
;3639           .SET/V065=<.OR[   <.EQL[<1OP/> ,<1OP/LOD>]>     ,   <.EQL[<CNT1.MEM/>     ,<CNT1.REG/>]>      ]>
;3640           .SET/V066=<VCNT0/>
;3641           .SET/V067=<VCNT1/>
;3642           .SET/V068=<VMEM/>
;3643           .SET/V069=<VREG/>
;3644
;3645   .TOC    "       Validity Checks                 : DSIZE Checks"
;3646           .SET/V070=<VSIZE/>
;3647           .SET/V071=<.OR[ <VSIZE/> , <.NEQ[<BUT/>,<BUT/CCBR>,<BUT/CCBR1.CCBR0.IR0>,<BUT/CCBR0.SRKSTA0>,<BUT/CCBR1.INT-TS>]>]>
;3648           .SET/V072=<.OR[ <VSIZE/> , <.NEQ[<BUT/>,<BUT/SRKSTA>,<BUT/CCBR0.SRKSTA0>]>]>
;3649
;3650   .TOC    "       Validity Checks                 : Multiple WBUS Drive Checks"   ;SEE CHARTS TO DECIPHER
;3651           .SET/V080=<.CASE[<WBUS.DRIVE>]OF[1,1,0,1,0,1,0,0,0,0,0,0,0,0,0,0]>
;3652           .SET/WBUS.DRIVE=<.SUM[<ALU.GROUP>,<WCTRL.GROUP>,<OTHERS>]>
;3653
;3654   .TOC    "       Validity Checks                 :       ALU Group"
;3655           .SET/ALU.GROUP=<.OR[  <.EQL[<LIT/>,<LIT/LONLIT>]>    ,   <WMUX>]>
;3656           .SET/WMUX =<.CASE[<MUX/>]OF[1,1,1,1,1,1,1,1,1,1,<MUX.09>,1,1,1,<MUX.0D>,1,1]>
;3657           .SET/MUX.09=<.CASE[<ALU/>]OF[1,1,1,1,1,1,1,1,1,0,0,<DQ.0>,<DQ.1>,0,0,<DQ.0>,<DQ.1>]>
;3658           .SET/MUX.0D=<.CASE[<ALU/>]OF[1,1,1,1,1,1,1,1,1,1,0,00000,00000,1,0,00001,00001]>
;3659           .SET/DQ.0=  <.CASE[<DQ1/>]OF[0,1,1,1]>
;3660           .SET/DQ.1=  <.CASE[<DQ1/>]OF[1,0,1,1]>
;3661
;3662   .TOC    "       Validity Checks                 :       WCTRL Group"
;3663           .SET/WCTRL.GROUP =<.CASE[<WCTRL/>]OF[0,0,0,3,3,0,0,<WCTRL.07>,0,0,0,0,3,3,3,3,  0,3,0,0,0,0,0,0,0,0,3,3,3,3,3,3,
;3664                                                0,0,0,0,0,0,0,0,          0,0,0,0,0,0,0,0,0,0,  0,0,3,0,0,0,0,0,0,0,3,0,0,0,0,3]>
;3665           .SET/WCTRL.07    =<.SELECT[  <.EQL[<CCMISC/>,<CCMISC/WB_ATCR.CCBR_SIGND>]>  ,  3]>
;3666
;3667   .TOC    "       Validity Checks                 :       Others"
;3668           .SET/OTHERS=<.OR[  <.CASE[<FPA/>]OF[0,0,0,0,5,5]>
;3669                              <.CASE[<MSRC/>]OF[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,5,0,0,0,0,0,5,0,5,0]>  ]>
;3670
;3671   .CREF                   ;END OF CREF ONLY WHEN FULL ASSEMBLY THE FOLLOWING WILL ALWAYS CREF
                                  ;3672  .BIN
```

3672; This page intentionally left blank.

```
                                        ;3673  .TOC "MACRO.MIC"
                                        ;3674  .TOC "REVISION 60.0"
                                        ;3675  ;     Jeff Peng, Gerard Koeckhoven, Brian Allison, C. E. MCDOWELL,  P. R. GUILBAULT
                                        ;3676
;3677    .NOBIN
;3678    .TOC "  Revision History"
;3679
;3680    ; 60    Add Macro in IANDE.MIC for restore data from M[ERRCOD]
;3681    ; 59    Add Macro for OSR.MIC increment PC
;3682    ;       Add Macro for setting patch bit
;3683    ; 58    Add Macro for interval timer
;3684    ; 57    Add Macros to support CMT053 (GH)
;3685    ; 56    Add Macro to support CMT052
;3686    ; 55    ADD MACROS TO SUPPORT CMT049
;3687    ; 54    Initial release.
```

```
;3688    .IOC "  Basic Macros"
;3689
;3690    CCOP1                           "CC/CCOP1.CCBR_SIGND"
;3691    CCOP2                           "CC/CCOP2.CCBR_SIGND"
;3692    CLEAR ADD1(FLAG0)               "MISC/CLR.FLAG0"
;3693    CLEAR ADD2(FLAG1)               "MISC/CLR.FLAG1"
;3694    CLEAR ARITH TRAPS               "CCMISC/WB_ATCR.CCBR_SIGND"
;3695    CLEAR BOOT(FLAG MMNOINT)        "MISC/CLR.MMNOINT"
;3696    CLEAR FLAG0                     "MISC/CLR.FLAG0"
;3697    CLEAR FLAG1                     "MISC/CLR.FLAG1"
;3698    CLEAR FLAG2                     "MISC/CLR.FLAG2"
;3699    CLEAR FLAG3                     "MISC/CLR.FLAG3"
;3700    CLEAR FLAG4                     "MISC/CLR.MMNOINT"
;3701    CLEAR FP TRAPS                  "WCTRL/FPTCR"
;3702    CLEAR FPA(FLAG0)                "MISC/CLR.FLAG0"
;3703    CLEAR FPD                       "MISC/CLR.FPD"
;3704    CLEAR GFLOAT(FLAG4)             "MISC/CLR.MMNOINT"
;3705    CLEAR MM.NOINT                  "MISC/CLR.MMNOINT"
;3706    CLEAR MOPZERO(FLAG1)            "MISC/CLR.FLAG1"
;3707    CLEAR MUL1(FLAG2)               "MISC/CLR.FLAG2"
;3708    CLEAR MUL2(FLAG3)               "MISC/CLR.FLAG3"
;3709    CLEAR OPZERO(FLAG3)             "MISC/CLR.FLAG3"
;3710    CLEAR OVER(FLAG2)               "MISC/CLR.FLAG2"
;3711    CLEAR POP1C(FLAG4)              "MISC/CLR.MMNOINT"
;3712    CLEAR READ(FLAG1)               "MISC/CLR.FLAG1"
;3713    CLEAR REGINT(FLAG1)             "MISC/CLR.FLAG1"
;3714    CLEAR SAMESIGN(FLAG4)           "MISC/CLR.MMNOINT"
;3715    CLEAR STACK FLAG                "MISC/CLR.STACKFLG"
;3716    CLEAR SUB(FLAG1)                "MISC/CLR.FLAG1"
;3717    CLEAR TP                        "MISC/CLR.TP"
;3718    CLEAR WRITE(FLAG1)              "MISC/CLR.FLAG1"
;3719    CLOBBER MTEMP0                  "MSRC/TEMP0,SPW/MLONG"
;3720    CLOBBER MTEMP0 DEF              "SPW/MLONG"
;3721
;3722    DEC STEPC                       "MISC/DEC.SC"
;3723    DIVDA SOR IN R[]                "ALPCTL/DIVDA,RSRC/@1,ROT/0"
;3724    DIVDS SOR IN R[]                "ALPCTL/DIVDS,RSRC/@1,ROT/0"
;3725    DIVFAST+ SOR IN R[]             "ALPCTL/DIVFAST+,RSRC/@1,ROT/0"
;3726    DIVFAST- SOR IN R[]             "ALPCTL/DIVFAST-,RSRC/@1,ROT/0"
;3727
;3728    FLUSH XB                        "WCTRL/PC_WB,WB_M[PC]"
;3729    FPAWAIT                         "LIT/FPAWAIT"
;3730    FORCE 32 BITS OF VA             "BUS/PRB.RD,VSIZE/1"
;3731    FORCE CACHE PARITY              "MISC/FORCE.CACHE,VSIZE/1"
;3732
;3733    IO RESET                        "BUS/IOINIT"
;3734    IRD1                            "BUT/IRD1,NEXT/3F9"     ; 3F9 = IE.IRD1.ERROR
;3735    IRD1TEST                        "BUT/IRD1TST"
;3736    IRDX []                         "BUT/IRDX,NEXT/@1"
;3737    ISIZE[]                         "ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/@1"
;3738
;3739    MULFAST+ CAND IN R[]            "ALPCTL/MULFAST+,RSRC/@1,ROT/0"
;3740    MULFAST- CAND IN R[]            "ALPCTL/MULFAST-,RSRC/@1,ROT/0"
;3741
;3742    NOP                             "ALPCTL/NOP"
```

```
;3743
;3744   PATCH                              ''PATCH/1''
;3745   PUSH                               ''JSR/PUSH''
;3746   PUSH RBS+                          'MSRC/PSHADD''
;3747   PUSH RBS-                          'MSRC/PSHSUB''
;3748
;3749   PROCESS INIT                       'BUS/PRINIT''
;3750
;3751   RESTORE POWER FAIL INT             'WCTRL/NOP''
;3752   RETURN []                          ''BUT/RETURN,NEXT/@1''
;3753   RETURN AND INHIBIT DESTINATIONS    ''BUT/RET.DINH''
;3754   RETURN AND SUPPRESS BUS CYCLE      ''BUT/RETURN,NEXT/0,MISC/RSBC''
;3755
;3756   SET ADD1(FLAG0)                    'MISC/SET.FLAG0''
;3757   SET ADD2(FLAG1)                    'MISC/SET.FLAG1''
;3758   SET BOOT(FLAG MMNOINT)             'MISC/SET.MMNOINT''
;3759   SET FLAG0                          'MISC/SET.FLAG0''
;3760   SET FLAG1                          'MISC/SET.FLAG1''
;3761   SET FLAG2                          'MISC/SET.FLAG2''
;3762   SET FLAG3                          'MISC/SET.FLAG3''
;3763   SET FLAG4                          'MISC/SET.MMNOINT''
;3764   SET FPA(FLAG0)                     'MISC/SET.FLAG0''
;3765   SET FPD                            'MISC/SET.FPD''
;3766   SET GFLOAT(FLAG4)                  'MISC/SET.MMNOINT''
;3767   SET MM.NOINT                       'MISC/SET.MMNOINT''
;3768   SET MOPZERO(FLAG1)                 'MISC/SET.FLAG1''
;3769   SET MUL1(FLAG2)                    'MISC/SET.FLAG2''
;3770   SET MUL2(FLAG3)                    'MISC/SET.FLAG3''
;3771   SET OPZERO(FLAG3)                  'MISC/SET.FLAG3''
;3772   SET OVER(FLAG2)                    'MISC/SET.FLAG2''
;3773   SET POP1C(FLAG4)                   'MISC/SET.MMNOINT''
;3774   SET READ(FLAG1)                    'MISC/SET.FLAG1''
;3775   SET REGINT(FLAG1)                  'MISC/SET.FLAG1''
;3776   SET SAMESIGN(FLAG4)                'MISC/SET.MMNOINT''
;3777   SET SIGN(FLAG0)                    'MISC/SET.FLAG0''
;3778   SET STACK FLAG                     'MISC/SET.STACKFLG''
;3779   SET SUB(FLAG1)                     'MISC/SET.FLAG1''
;3780   SET UNDER(FLAG3)                   'MISC/SET.FLAG3''
;3781   SET V                              ''CCMISC/SETV.CCBR_SIGND''
;3782   SET WRITE(FLAG1)                   'MISC/SET.FLAG1''
;3783   SIZE[]                             ''VSIZE/1,DTYPE/@1''
```

```
;3784    .TOC "  Bus Function Macros"
;3785
;3786    COMPLETE CPU BUS CYCLES          "BUS/PRB.RD.PTE.K"
;3787
;3788    READ                             "BUS/READ"
;3789    READ.LONG                        "BUS/READ.LNG"
;3790    READ.LONG.MOD                    "BUS/READ.LNG.MOD"
;3791    READ.MOD                         "BUS/READ.MOD"
;3792    READ.MOD.LOCK                    "BUS/READ.MOD.LCK"
;3793    READ.NOTRAP                      "BUS/READ.NT"
;3794    READ.PHY                         "BUS/READ.PHY"
;3795    READ.SECOND                      "BUS/READ.SEC"
;3796
;3797    WRITE                            "BUS/WRITE,WCTRL/WDR_WB"
;3798    WRITE (M[] R[]).RR.4             "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,RSRC/@2,ALPCTL/WX_S,ROT/RR.MR.4"
;3799    WRITE -M[]                       "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,RSRC/ZERO,ALU/B-A-CI,ALUCI/ZERO,MUX/M.R1"
;3800    WRITE -Q                         "BUS/WRITE,WCTRL/WDR_WB,MUX/R.Q,RSRC/ZERO,ALU/A-B-CI,ALUCI/ZERO"
;3801    WRITE CVTNP(M[])                 "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,ALU/A+B+CI.BCD,MUX/R.S,RSRC/ZERO,ROT/CVTNP"
;3802    WRITE CVTPN(M[])                 "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,RSRC/TEMP0,ALPCTL/WX_S,ROT/CVTPN"
;3803    WRITE D+R[]+ALKC                 "BUS/WRITE,WCTRL/WDR_WB,RSRC/@1,MUX/D.R1,ALU/A+B+CI,ALUCI/ALKC"
;3804    WRITE D.OR.ZLIT28[]              "BUS/WRITE,WCTRL/WDR_WB,MUX/D.S,ROT/ZLIT28,LIT/LITRL,LITRL/@1,ALU/OR"
;3805    WRITE M[]                        "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,ALU/OR,MUX/M.S,ROT/ZERO"
;3806    WRITE M[]+PSLC                   "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/A+B+CI,ALUCI/PSLC"
;3807    WRITE M[]+Q                      "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,MUX/M.Q1,ALU/A+B+CI"
;3808    WRITE M[]+Q+PSLC                 "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,MUX/M.Q1,ALU/A+B+CI,ALUCI/PSLC"
;3809    WRITE M[]-PSLC                   "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/A-B-CI,ALUCI/PSLC"
;3810    WRITE M[]-Q                      "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,MUX/M.Q1,ALU/A-B-CI"
;3811    WRITE M[]-Q-PSLC                 "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,MUX/M.Q1,ALU/A-B-CI,ALUCI/PSLC"
;3812    WRITE M[].AND.ZLIT0[]            "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/AND"
;3813    WRITE M[].ANDNOT.Q               "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,MUX/M.Q1,ALU/ANDNOT"
;3814    WRITE M[].ANDNOT.R[]             "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,RSRC/@2,ALU/ANDNOT,MUX/M.R1"
;3815    WRITE M[].ANDNOT.ZLIT8[]         "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT8,MUX/M.S,ALU/ANDNOT"
;3816    WRITE M[].OR.Q                   "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,MUX/M.Q1,ALU/OR"
;3817    WRITE M[].OR.R[]                 "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,RSRC/@2,ALU/OR,MUX/M.R1"
;3818    WRITE M[].OR.ZLIT0[]             "BUS/WRITE,WCTRL/WDR_WB,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2"
;3819    WRITE M[].OR.ZLIT28[]            "BUS/WRITE,WCTRL/WDR_WB,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZLIT28,LIT/LITRL,LITRL/@2"
;3820    WRITE M[].RR.P                   "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,ROT/RR.MM.P,ALPCTL/WX_S"
;3821    WRITE M[].SL.1                   "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,ROT/ZERO,MUX/M.S,ALU/A+B+CI.SL"
;3822    WRITE M[].XOR.Q                  "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,MUX/M.Q1,ALU/XOR"
;3823    WRITE M[].XZ                     "BUS/WRITE,WCTRL/WDR_WB,MSRC/@1,ALPCTL/WX_S,ROT/XZ.MM"
;3824    WRITE NOTREG                     "BUS/WRITE.NOREG,WCTRL/WDR_WB"
;3825    WRITE Q                          "BUS/WRITE,WCTRL/WDR_WB,RSRC/ZERO,MUX/R.Q,ALU/OR"
;3826    WRITE Q.NOT                      "BUS/WRITE,WCTRL/WDR_WB,RSRC/ZERO,MUX/R.Q,ALU/A-B-CI,ALUCI/ONE"
;3827    WRITE Q (Q.SL.1).OR.1            "BUS/WRITE,WCTRL/WDR_WB,DQ1/Q_WX,ALU/A+B+CI.SL,MUX/R.Q,RSRC/ZERO,ALUSHF/ONE"
;3828    WRITE R[]                        "BUS/WRITE,WCTRL/WDR_WB,RSRC/@1,ALU/OR,MUX/R.S,ROT/ZERO"
;3829    WRITE R[]+CONX(4)                "BUS/WRITE,WCTRL/WDR_WB,RSRC/@1,ALU/A+B+CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG"
;3830    WRITE R[]-D-ALKC                 "BUS/WRITE,WCTRL/WDR_WB,RSRC/@1,MUX/D.R1,ALU/B-A-CI,ALUCI/ALKC"
;3831    WRITE R[]-M[]                    "BUS/WRITE,WCTRL/WDR_WB,MSRC/@2,RSRC/@1,ALU/B-A-CI,MUX/M.R1"
;3832    WRITE R[]-M[]-1                  "BUS/WRITE,WCTRL/WDR_WB,MSRC/@2,RSRC/@1,ALU/B-A-CI,MUX/M.R1,ALUCI/ONE"
;3833    WRITE XB PC_PC+1                 "BUS/WRITE,WCTRL/WDR_WB,MSRC/XB.PC_PC+1,ROT/ZERO,ALU/OR,MUX/M.S,
;3834                                                ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/BYTE"
;3835    WRITE XB PC_PC+4                 "BUS/WRITE,WCTRL/WDR_WB,MSRC/XB.PC_PC+1,ROT/ZERO,ALU/OR,MUX/M.S,
;3836                                                ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/LONG"
;3837    WRITE ZLIT0[]                    "BUS/WRITE,WCTRL/WDR_WB,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@1"
;3838    WRITE.LONG                       "BUS/WRITE.LNG,WCTRL/WDR_WB.UR"
```

```
;3839   WRITE.LONG D                    'BUS/WRITE.LNG,WCTRL/WDR_WB.UR,RSRC/ZERO,MUX/D.R1,ALU/OR''
;3840   WRITE.LONG M[].ANDNOT.Q         'BUS/WRITE.LNG,WCTRL/WDR_WB.UR,MSRC/@1,MUX/M.Q1,ALU/ANDNO '
;3841   WRITE.LONG M[].OR.Q             'BUS/WRITE.LNG,WCTRL/WDR_WB.UR,MSRC/@1,MUX/M.Q1,ALU/OR''
;3842   WRITE.LONG.NOTRAP               'BUS/WRITE.NT.LNG''
;3843   WRITE.NOTRAP                    'BUS/WRITE.NT''
;3844   WRITE.PHY                       'BUS/WRITE.PHY''
;3845   WRITE.PHY M[]                   'BUS/WRITE.PHY,WCTRL/WDR_WB,MSRC/@1,ALU/OR,MUX/M.S,ROT/ZERO''
;3846   WRITE.PHY R[]                   'BUS/WRITE.PHY,WCTRL/WDR_WB,RSRC/@1,ALU/OR,MUX/R.S,ROT/ZERO''
;3847   WRITE.SECOND                    'BUS/WRITE.SEC''
;3848   WRITE.SECOND.UL                 'BUS/WRITE.UL.SEC''
;3849   WRITE.UL M[]                    'BUS/WRITE.UL,WCTRL/WDR_WB,MSRC/@1,ALU/OR,MUX/M.S,ROT/ZERO''
;3850   WRITE.UL M[]+Q                  'BUS/WRITE.UL,WCTRL/WDR_WB,MSRC/@1,MUX/M.Q1,ALU/A+B+CI''
;3851   WRITE.UL M[].ANDNOT.Q           'BUS/WRITE.UL,WCTRL/WDR_WB.UR,MSRC/@1,MUX/M.Q1,ALU/ANDNOT''
;3852   WRITE.UL M[].OR.Q               'BUS/WRITE.UL,WCTRL/WDR_WB.UR,MSRC/@1,MUX/M.Q1,ALU/OR''
;3853   WRITE.UL M[].OR.ZLITO[]         'BUS/WRITE.UL,WCTRL/WDR_WB,MSRC/@1,ALU/OR,MUX/M.S,ROT/ZLITO,LIT/LITRL,LITRL/@2''
;3854   WRITE.UL M[].ANDNOT.ZLITO[]     'BUS/WRITE.UL,WCTRL/WDR_WB,MSRC/@1,ALU/ANDNOT,MUX/M.S,ROT/ZLITO,LIT/LITRL,LITRL/@2''
```

```
3855    .TOC "  Register Transfer Macros"
3856
3857    ALUS_BCD SIGN.ZERO              "CCMISC/ALUS_DSDZ.CCBR_ALUS"
3858    ALUS_BCD SIGN.ZERO(M[])         "CCMISC/ALUS_DSDZ.CCBR_ALUS,MSRC/@1,RSRC/ZERO,ALU/OR,MUX/M.R1"
3859    ALUS_SIGND                      "CCMISC/ALUS_SIGND.CCBR_ALUS"
3860    ALUS_UNSGN                      "CCMISC/ALUS_UNSGN.CCBR_ALUS"
3861    ASTLVL_M[].RL.24                "WCTRL/ASTLVL_WB,ALPCTL/WX_S,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE"
3862    ASTLVL_R[]_M[]                  "WCTRL/ASTLVL_WB,SPW/RLONG,RSRC/@1,ALU/OR,MUX/M.S,ROT/ZERO,MSRC/@2"
3863    ASTLVL_[]                       "WCTRL/ASTLVL_WB,LITRL/@1,LIT/LITRL,ROT/ZLIT24,ALPCTL/WX_S"
3864
3865    BUS GRANT M[]_IPL               "BUS/GRANT,WCTRL/GRANT,SPW/MLONG,MSRC/@1"
3866
3867    CC_FPA                          "CCPSL/CC_WB.CCBR_ALUS,FPA/WBUS_FPA.CC"
3868    CC_M[]                          "CCPSL/CC_WB.CCBR_ALUS,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZERO"
3869    CC_M[].NOTAND.R[]               "CCPSL/CC_WB.CCBR_ALUS,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/NOTAND"
3870    CC_M[].OR.R[]                   "CCPSL/CC_WB.CCBR_ALUS,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/OR"
3871    CC_M[].OR.ZLIT0[]               "CCPSL/CC_WB.CCBR_ALUS,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/OR"
3872    CC_M[].XOR.ZLIT0[]              "CCPSL/CC_WB.CCBR_ALUS,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/XOR"
3873    CC_M[]_MB.AND.ZLIT0[]           "CCPSL/CC_WB.CCBR_ALUS,MSRC/@1,SPW/MLONG,ALU/AND,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2"
3874    CC_M[]_MB.ANDNOT.CONX(1)        "CCPSL/CC_WB.CCBR_ALUS,ALU/ANDNOT,MUX/M.S,SPW/MLONG,MSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE"
3875    CC_M[]_MB.ANDNOT.CONX(4)        "CCPSL/CC_WB.CCBR_ALUS,ALU/ANDNOT,MUX/M.S,SPW/MLONG,MSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG"
3876    CC_M[]_MB.OR.CONX(1)            "CCPSL/CC_WB.CCBR_ALUS,ALU/OR,MUX/M.S,SPW/MLONG,MSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE"
3877    CC_M[]_ZLIT0[]                  "CCPSL/CC_WB.CCBR_ALUS,SPW/MLONG,MSRC/@1,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@2"
3878    CC_R[]                          "CCPSL/CC_WB.CCBR_ALUS,ALU/OR,MUX/R.S,RSRC/@1,ROT/ZERO"
3879    CC_ZLIT0[]                      "CCPSL/CC_WB.CCBR_ALUS,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@1"
3880    CC_[]                           "CCPSL/CC_WB.CCBR_ALUS,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@1"
3881
3882    CONREGS_D_M[]_R[]               "WCTRL/CONWRITE,MSRC/@1,SPW/MLONG,ALU/OR,MUX/R.S,ROT/ZERO,RSRC/@2,DQ1/D_WX"
3883    CONREGS_M[]                     "WCTRL/CONWRITE,WB_M[@1]"
3884    CONREGS_M[].OR.ZLIT16[]         "WCTRL/CONWRITE,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZLIT16,LIT/LITRL,LITRL/@2"
3885    CONREGS_M[].RR.16               "WCTRL/CONWRITE,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,MSRC/@1,ALPCTL/WX_S"
3886    CONREGS_M[]_R[]                 "WCTRL/CONWRITE,MSRC/@1,SPW/MLONG,RSRC/@2,MUX/R.S,ALU/OR,ROT/ZERO"
3887    CONREGS_R[]                     "WCTRL/CONWRITE,RSRC/@1,MUX/R.S,ALU/OR,ROT/ZERO"
3888    CONREGS_ZLIT16[]                "WCTRL/CONWRITE,ALPCTL/WX_S,ROT/ZLIT16,LIT/LITRL,LITRL/@1"
3889
3890    CRAR_ZLIT16[]                   "WCTRL/LOADCRAR,LITRL/@1,LIT/LITRL,ROT/ZLIT16,ALPCTL/WX_S"
3891
3892    D(OD)_ZLIT0[]                   "DQ1/D_WX,ROT/ZLIT0,LIT/LITRL,LITRL/@1,ALUOD/OR.OD,MUX/Z.S"
3893    D_(M[] R[]).RR.9                "DQ1/D_WX,ALPCTL/WX_D_S,MSRC/@1,RSRC/@2,ROT/RR.MR.9"
3894    D_(M[] R[]).RR.P                "ALPCTL/WX_D_S,MSRC/@1,RSRC/@2,ROT/RR.MR.P"
3895    D_(M[]+CONX(2)).SR.1            "DQ1/D_WX,ALU/A+B+CI.SR,MUX/M.S,MSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD"
3896    D_(M[].RR.P).AND.R[]            "DQ1/D_WX,MSRC/@1,RSRC/@2,ROT/RR.MM.P,ALU/AND,MUX/R.S"
3897    D_(R[] M[]).RL.P                "ALPCTL/WX_D_S,MSRC/@2,RSRC/@1,ROT/RL.RM.P"
3898    D_(R[]+CONX(2)).SR.1            "DQ1/D_WX,ALU/A+B+CI.SR,MUX/R.S,RSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD"
3899    D_-1                            "ALPCTL/WX_D_S,ROT/MINUS1"
3900    D_CONX.SIZ                      "ALPCTL/WX_D_S,ROT/CONX.SIZ"
3901    D_D+OLIT24[]                    "DQ1/D_WX,MUX/D.S,ALU/A+B+CI,ROT/OLIT24,LIT/LITRL,LITRL/@1"
3902    D_D+R[]                         "DQ1/D_WX,RSRC/@1,MUX/D.R1,ALU/A+B+CI"
3903    D_D+R[]+ALKC                    "DQ1/D_WX,RSRC/@1,MUX/D.R1,ALU/A+B+CI,ALUCI/ALKC"
3904    D_D+ZLIT0[]                     "DQ1/D_WX,MUX/D.S,ALU/A+B+CI,ROT/ZLIT0,LIT/LITRL,LITRL/@1"
3905    D_D-1                           "DQ1/D_WX,ALU/A+B+CI,MUX/D.S,ROT/MINUS1"
3906    D_D-CONX(2)                     "DQ1/D_WX,ALU/A-B-CI,MUX/D.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD"
3907    D_D-CONX(4)                     "DQ1/D_WX,ALU/A-B-CI,MUX/D.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG"
3908    D_D-R[]                         "DQ1/D_WX,RSRC/@1,MUX/D.R1,ALU/A-B-CI"
3909    D_D-ZLIT0[]                     "DQ1/D_WX,LIT/LITRL,LITRL/@1,ROT/ZLIT0,MUX/D.S,ALU/A-B-CI"
```

```
;3910   D_D.AND.OLIT24[]                 ''DQ1/D_WX,ALU/AND,MUX/D.S,ROT/OLIT24,LIT/LITRL,LITRL/@1''
;3911   D_D.AND.ZLIT0[]                  ''DQ1/D_WX,ALU/AND,MUX/D.S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;3912   D_D.AND.ZLIT28[]                 ''DQ1/D_WX,ALU/AND,MUX/D.S,ROT/ZLIT28,LIT/LITRL,LITRL/@1''
;3913   D_D.OR.R[]                       ''DQ1/D_WX,ALU/OR,MUX/D.R1,RSRC/@1''
;3914   D_D.XOR.Q                        ''DQ1/D_WX,ALU/XOR,MUX/D.Q1''
;3915   D_D.XOR.ZLIT12[]                 ''DQ1/D_WX,MUX/D.S,ALU/XOR,ROT/ZLIT12,LIT/LITRL,LITRL/@1''
;3916   D_M[]                            ''DQ1/D_WX,MSRC/@1,RSRC/ZERO,ALU/OR,MUX/M.R1''
;3917   D_M[] SQR                        ''DQ2/SQR.D_WX,MSRC/@1,RSRC/ZERO,ALU/OR,MUX/M.R2''
;3918   D_M[]+R[]                        ''DQ1/D_WX,MSRC/@1,RSRC/@2,ALU/A+B+CI,MUX/M.R1''
;3919   D_M[]+ZLIT0[]                    ''DQ1/D_WX,ALU/A+B+CI,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;3920   D_M[]-R[]                        ''DQ1/D_WX,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/A-B-CI''
;3921   D_M[]-ZLIT0[]                    ''DQ1/D_WX,ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;3922   D_M[].AND.R[]                    ''DQ1/D_WX,ALU/AND,MUX/M.R1,MSRC/@1,RSRC/@2''
;3923   D_M[].AND.ZLIT16[]               ''DQ1/D_WX,ALU/AND,MUX/M.S,MSRC/@1,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;3924   D_M[].OR.(R[].RL.24)             ''DQ1/D_WX,ALU/OR,MUX/M.S,MSRC/@1,ROT/RR.RR.SIZ,RSRC/@2,DTYPE/BYTE,VSIZE/1''
;3925   D_M[].OR.R[]                     ''DQ1/D_WX,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/OR''
;3926   D_M[].RR.16                      ''ALPCTL/WX_D_S,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD''
;3927   D_M[].RR.16 Q_R[]                ''ALPCTL/WX_D_S.Q_R,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,RSRC/@2''
;3928   D_M[].SR.1                       ''DQ1/D_WX,ALU/A+B+CI.SR,MUX/M.S,MSRC/@1,ROT/ZERO''
;3929   D_Q Q_D                          ''ALPCTL/WX_D_Q.Q_D''
;3930   D_Q_-1                           ''ALPCTL/WX_D_Q_S,ROT/MINUS1''
;3931   D_Q_0                            ''ALPCTL/WX_D_Q_S,ROT/ZERO''
;3932   D_Q_D.XOR.R[]                    ''DQ1/Q_D_WX,ALU/XOR,MUX/D.R1,RSRC/@1''
;3933   D_Q_M[]                          ''DQ1/Q_D_WX,MSRC/@1,ROT/ZERO,MUX/M.S,ALU/OR''
;3934   D_Q_M[].XOR.Q                    ''DQ1/Q_D_WX,ALU/XOR,MUX/M.Q1,MSRC/@1''
;3935   D_RNUM_D+ZLIT0[]                 ''DQ1/D_WX,MSRC/RNUM_WBUS,ALU/A+B+CI,MUX/D.S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;3936   D_RNUM_D-ZLIT0[]                 ''DQ1/D_WX,MSRC/RNUM_WBUS,ALU/A-B-CI,MUX/D.S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;3937   D_RNUM_ZLIT0[]                   ''ALPCTL/WX_D_S,MSRC/RNUM_WBUS,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;3938   D_R[]                            ''DQ1/D_WX,ALU/OR,MUX/R.S,RSRC/@1,ROT/ZERO''
;3939   D_R[]+CONX.SIZ                   ''DQ1/D_WX,RSRC/@1,ROT/CONX.SIZ,MUX/R.S,ALU/A+B+CI''
;3940   D_R[]-1                          ''DQ1/D_WX,ALU/A+B+CI,MUX/R.S,RSRC/@1,ROT/MINUS1''
;3941   D_R[]-D-ALKC                     ''DQ1/D_WX,RSRC/@1,MUX/D.R1,ALU/B-A-CI,ALUCI/ALKC''
;3942   D_R[]-M[]                        ''DQ1/D_WX,RSRC/@1,MSRC/@2,MUX/M.R1,ALU/B-A-CI''
;3943   D_R[].OR.Q                       ''DQ1/D_WX,RSRC/@1,MUX/R.Q,ALU/OR''
;3944   D_R[].SR.1                       ''DQ1/D_WX,ALU/A+B+CI.SR,MUX/R.S,RSRC/@1,ROT/ZERO''
;3945   D_R[]_ZEXT(XB) PC_PC+1           ''DQ1/D_WX,RSRC/@1,SPW/RLONG,MSRC/XB.PC_PC+1,ROT/ZERO,MUX/XM.S,ALU/OR,
;3946                                              ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/BYTE''
;3947   D_SEXT(XB) PC_PC+1               ''DQ1/D_WX,MSRC/XB.PC_PC+1,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/OR,VSIZE/1,DTYPE/BYTE,
;3948                                              ISTRM/ISIZE_DSIZE''
;3949   D_SEXT(XB) PC_PC+2               ''DQ1/D_WX,MSRC/XB.PC_PC+1,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/OR,VSIZE/1,DTYPE/WORD,
;3950                                              ISTRM/ISIZE_DSIZE''
;3951   D_ZEXT(M[])                      ''DQ1/D_WX,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALUXM/ZERO,ALU/OR''
;3952   D_ZLIT0[]                        ''ALPCTL/WX_D_S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;3953   D_ZLIT0[]-D                      ''DQ1/D_WX,MUX/D.S,ALU/B-A-CI,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;3954   D_ZLIT12[]                       ''ALPCTL/WX_D_S,ROT/ZLIT12,LIT/LITRL,LITRL/@1''
;3955   D_ZLIT16[]                       ''ALPCTL/WX_D_S,ROT/ZLIT16,LIT/LITRL,LITRL/@1''
;3956   D_ZLIT24[]                       ''ALPCTL/WX_D_S,ROT/ZLIT24,LIT/LITRL,LITRL/@1''
;3957   D_ZLIT24[] M[]_HARD.REV          ''DQ1/D_WX,ALUOD/OR.OD,MUX/Z.S,ROT/ZLIT24,LIT/LITRL,LITRL/@1,MSRC/@2,SPW/MLONG,
;3958                                              WCTRL/REVLEVEL''
;3959   D_ZLIT4[]                        ''ALPCTL/WX_D_S,ROT/ZLIT4,LIT/LITRL,LITRL/@1''
;3960   D_ZLIT8[]                        ''ALPCTL/WX_D_S,ROT/ZLIT8,LIT/LITRL,LITRL/@1''
;3961
;3962   FLAGS_D_R[]                      ''WCTRL/FLAGS_WB,RSRC/@1,ROT/ZERO,ALU/OR,MUX/R.S,DQ1/D_WX''
;3963   FLAGS_M[].AND.ZLIT0[]            ''WCTRL/FLAGS_WB,MSRC/@1,ALU/AND,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;3964   FLAGS_R[]                        ''WCTRL/FLAGS_WB,RSRC/@1,ROT/ZERO,ALU/OR,MUX/R.S''
```

```
;3965   FLAGS_ZLIT0[]                    'WCTRL/FLAGS_WB,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;3966
;3967   FPA.ENABLE_M[].RR.P               'WCTRL/FPA.ENABLE_WB5,ALPCTL/WX_S,ROT/RR.MM.P,MSRC/@1''
;3968   FPA_MB_M[]_R[]                   'FPA/FPA_DATA.MBUS,MSRC/@1,SPW/MLONG,RSRC/@2,MUX/R.S,ROT/ZERO,ALU/OR''
;3969   FPA_M[]                          'FPA/FPA_DATA.MBUS,MSRC/@1''
;3970   FPA_M[] FPA_WB_R[]-0             'FPA/FPA_MBUS.FPA_WBUS,MSRC/@1,RSRC/@2,MUX/R.S,ALU/A-B-CI,ROT/ZERO''
;3971   FPA_M[] MDR_R[]                  'FPA/FPA_DATA.MBUS,MSRC/@1,WCTRL/MDR_WB,ALU/OR,MUX/R.S,ROT/ZERO,RSRC/@2''
;3972   FPA_Q_MDR MTEMP0_R[]             'FPA/FPA_DATA.MBUS,SPW/MLONG,MSRC/MDR,RSRC/@1,ALPCTL/WX_R.Q_M''
;3973   FPA_Q_M[]                        'FPA/FPA_DATA.MBUS,MSRC/@1,MUX/M.S,ALU/OR,ROT/ZERO,DQ1/Q_WX''
;3974   FPA_Q_M[] MDR_Q                  'FPA/FPA_DATA.MBUS,WCTRL/MDR_WB,MSRC/@1,ALPCTL/WX_Q.Q_M''
;3975   FPA_Q_M[] MDR_R[]                'FPA/FPA_DATA.MBUS,ALPCTL/WX_R.Q_M,WCTRL/MDR_WB,MSRC/@1,RSRC/@2''
;3976   FPA_Q_M[].LITNXT                 'FPA/FPA_MBUS.LITNXT,MSRC/@1,MUX/M.S,ALU/OR,ROT/ZERO,DQ1/Q_WX''
;3977   FPA_Q_M[] VA_R[]                 'FPA/FPA_DATA.MBUS,ALPCTL/WX_R.Q_M,MSRC/@1,RSRC/@2,WCTRL/VA_WB''
;3978   FPA_R[].SIZ_M[]                  'FPA/FPA_DATA.MBUS,RSRC/@1,SPW/RSIZE,ALU/OR,MUX/M.S,ROT/ZERO,MSRC/@2''
;3979   FPA_R[]_M[]                      'FPA/FPA_DATA.MBUS,RSRC/@1,MSRC/@2,ROT/ZERO,SPW/RLONG,MUX/M.S,ALU/OR''
;3980   FPA_WB_R[]-0                     'FPA/FPA_DATA.WBUS,RSRC/@1,MUX/R.S,ALU/A-B-CI,ROT/ZERO''
;3981
;3982   INIR_M[].Q                       'WCTRL/INIR_WB,MSRC/@1,SPW/MLONG,ALU/OR,MUX/R.Q,RSRC/ZERO''
;3983   IPL_M[].RL.16                    'WCTRL/IPL_WB,ALPCTL/WX_S,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD''
;3984   IPL_[]                           'WCTRL/IPL_WB,ALPCTL/WX_S,ROT/ZLIT16,LIT/LITRL,LITRL/@1''
;3985
;3986   LONLIT_[]                        ''LIT/LONLIT,LONLIT/<.NOT[<LONLIT/@1>]>''
;3987
;3988   MB_M[]                           'MSRC/@1''
;3989   MDR_(M[] R[]).RR.9               'WCTRL/MDR_WB,MSRC/@1,RSRC/@2,ROT/RR.MR.9,ALPCTL/WX_S''
;3990   MDR_(M[] R[]).RR.P               'WCTRL/MDR_WB,MSRC/@1,RSRC/@2,ROT/RR.MR.P,ALPCTL/WX_S''
;3991   MDR_-1                           'WCTRL/MDR_WB,ROT/MINUS1,ALPCTL/WX_S''
;3992   MDR_-M[]                         'WCTRL/MDR_WB,MSRC/@1,ALU/B-A-CI,ALUCI/ZERO,RSRC/ZERO,MUX/M.R1''
;3993   MDR_0                            'WCTRL/MDR_0''
;3994   MDR_M[]                          'WCTRL/MDR_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/OR''
;3995   MDR_M[]+ALKC                     'WCTRL/MDR_WB,MSRC/@1,ALU/A+B+CI,ALUCI/ALKC,RSRC/ZERO,MUX/M.R1''
;3996   MDR_M[]+CONX(1)                  'WCTRL/MDR_WB,MSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE,ALU/A+B+CI,MUX/M.S''
;3997   MDR_M[]+R[]+ALKC                 'WCTRL/MDR_WB,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/A+B+CI,ALUCI/ALKC''
;3998   MDR_M[]-CONX.SIZ                 'WCTRL/MDR_WB,MSRC/@1,ALU/A-B-CI,ROT/CONX.SIZ,MUX/M.S''
;3999   MDR_M[].AND.OLIT8[]              'WCTRL/MDR_WB,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/OLIT8,MUX/M.S,ALU/AND''
;4000   MDR_M[].AND.ZLIT0[]              'WCTRL/MDR_WB,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/AND''
;4001   MDR_M[].ANDNOT.R[]               'WCTRL/MDR_WB,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/ANDNOT''
;4002   MDR_M[].ANDNOT.ZLIT0[]           'WCTRL/MDR_WB,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/ANDNOT''
;4003   MDR_M[].ASR.P                    'WCTRL/MDR_WB,MSRC/@1,ROT/ASR.M.P,ALPCTL/WX_S''
;4004   MDR_M[].FPLIT                    'WCTRL/MDR_WB,MSRC/@1,ROT/FPLIT,ALPCTL/WX_S''
;4005   MDR_M[].OR.(R[].RR.24)           'WCTRL/MDR_WB,MSRC/@1,RSRC/@2,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/LONG,MUX/M.S,ALU/OR''
;4006   MDR_M[].OR.CVTNP(R[])            'WCTRL/MDR_WB,MSRC/@1,RSRC/@2,ROT/CVTNP,ALU/OR,MUX/M.S''
;4007   MDR_M[].OR.R[]                   'WCTRL/MDR_WB,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/OR''
;4008   MDR_M[].OR.ZLIT24[]              'WCTRL/MDR_WB,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;4009   MDR_M[].RL.16                    'WCTRL/MDR_WB,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,ALPCTL/WX_S''
;4010   MDR_M[].RL.24                    'WCTRL/MDR_WB,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE,ALPCTL/WX_S''
;4011   MDR_M[].RL.8                     'WCTRL/MDR_WB,MSRC/@1,VSIZE/1,DTYPE/LONG,ROT/RR.MM.SIZ,ALPCTL/WX_S''
;4012   MDR_M[].RL.9                     'WCTRL/MDR_WB,ALPCTL/WX_S,ROT/RL.MM.PTE,MSRC/@1''
;4013   MDR_M[].RR.16                    'WCTRL/MDR_WB,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,ALPCTL/WX_S''
;4014   MDR_M[].XOR.R[]                  'WCTRL/MDR_WB,MSRC/@1,RSRC/@2,ALU/XOR,MUX/M.R1''
;4015   MDR_M[].XOR.ZLIT12[]             'WCTRL/MDR_WB,MSRC/@1,ROT/ZLIT12,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/XOR''
;4016   MDR_M[]_R[]                      'WCTRL/MDR_WB,MSRC/@1,SPW/MLONG,RSRC/@2,ROT/ZERO,MUX/R.S,ALU/OR''
;4017   MDR_M[]_R[].RR.16                'WCTRL/MDR_WB,MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/WORD,ALPCTL/WX_S''
;4018   MDR_M[]_ZLIT0[]                  'WCTRL/MDR_WB,MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT0,ALPCTL/WX_S''
;4019   MDR_Q                            'WCTRL/MDR_WB,RSRC/ZERO,MUX/R.Q,ALU/OR''
```

```
;4020   MDR_Q Q_M[]                        'WCTRL/MDR_WB,MSRC/@1,ALPCTL/WX_Q.Q_M''
;4021   MDR_Q_M[]                          'WCTRL/MDR_WB,DQ1/Q_WX,MSRC/@1,ROT/ZERO,MUX/M.S,ALU/OR''
;4022   MDR_Q_M[]-ZLIT8[]                 'WCTRL/MDR_WB,DQ1/Q_WX,MSRC/@1,ALU/A-B-CI,MUX/M.S,ROT/ZLIT8,LIT/LITRL,LITRL/@2''
;4023   MDR_R[]                           'WCTRL/MDR_WB,RSRC/@1,ROT/ZERO,MUX/R.S,ALU/OR''
;4024   MDR_R[]-CONX(1)                   'WCTRL/MDR_WB,RSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE,ALU/A-B-CI,MUX/R.S''
;4025   MDR_R[]-M[]-ALKC                  'WCTRL/MDR_WB,MSRC/@2,RSRC/@1,ALU/B-A-CI,ALUCI/ALKC,MUX/M.R1''
;4026   MDR_R[].RR.24                     'WCTRL/MDR_WB,RSRC/@1,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/LONG,ALPCTL/WX_S''
;4027   MDR_R[]_M[]                       'WCTRL/MDR_WB,RSRC/@1,SPW/RLONG,MSRC/@2,ALU/OR,MUX/M.S,ROT/ZERO''
;4028   MDR_R[]_RB-CONX.SIZ               'WCTRL/MDR_WB,RSRC/@1,ROT/CONX.SIZ,MUX/R.S,ALU/A-B-CI,SPW/RLONG,VSIZE/1,DTYPE/IDEP''
;4029   MDR_SEXT(M[])                     'WCTRL/MDR_WB,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/OR''
;4030   MDR_SEXT(XB)+R[] PC_PC+1          'WCTRL/MDR_WB,RSRC/@1,MSRC/XB.PC_PC+1,MUX/XM.R,ALUXM/SIGN,ALU/A+B+CI''
;4031   MDR_XB PC_PC+1                    'WCTRL/MDR_WB,MSRC/XB.PC_PC+1,RSRC/ZERO,MUX/M.R1,ALU/OR,ISTRM/ISIZE_DSIZE,
;4032                                                 VSIZE/1,DTYPE/BYTE''
;4033   MDR_XB PC_PC+2                    'WCTRL/MDR_WB,MSRC/XB.PC_PC+1,RSRC/ZERO,MUX/M.R1,ALU/OR,ISTRM/ISIZE_DSIZE,
;4034                                                 VSIZE/1,DTYPE/WORD''
;4035   MDR_XB PC_PC+4                    'WCTRL/MDR_WB,MSRC/XB.PC_PC+1,RSRC/ZERO,MUX/M.R1,ALU/OR,ISTRM/ISIZE_DSIZE,
;4036                                                 VSIZE/1,DTYPE/LONG''
;4037   MDR_XB PC_PC+1                    'WCTRL/MDR_WB,MSRC/XB.PC_PC+1,RSRC/ZERO,MUX/M.R1,ALU/OR,ISTRM/ISIZE_DSIZE,
;4038                                                 VSIZE/1,DTYPE/IDEP''
;4039   MDR_ZEXT(IR)                      'WCTRL/MDR_IR''
;4040   MDR_ZEXT(M[])                     'WCTRL/MDR_WB,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALUXM/ZERO,ALU/OR''
;4041   MDR_ZEXT(OSR)                     ''CCPSL/MDR_OSR.CCBR_BRATST''
;4042   MDR_ZLIT0[]                       'WCTRL/MDR_WB,LIT/LITRL,LITRL/@1,ROT/ZLIT0,ALPCTL/WX_S''
;4043   MDR_ZLIT16[]                      'WCTRL/MDR_WB,LIT/LITRL,LITRL/@1,ROT/ZLIT16,ALPCTL/WX_S''
;4044   MDR_ZLIT24[]                      'WCTRL/MDR_WB,LIT/LITRL,LITRL/@1,ROT/ZLIT24,ALPCTL/WX_S''
;4045
;4046   MEMSCAR_0_M[]_ZLIT0[]             'WCTRL/MEMSCAR_WB,MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4047   MEMSCAR_M[]-ZLIT0[]               'WCTRL/MEMSCAR_WB,MSRC/@1,ALU/A-B-CI,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4048   MEMSCAR_M[]_0                     'WCTRL/MEMSCAR_WB,MSRC/@1,SPW/MLONG,ROT/ZERO,ALPCTL/WX_S''
;4049   MEMSCAR_M[]_MB-ZLIT0[]            'WCTRL/MEMSCAR_WB,MSRC/@1,SPW/MLONG,ALU/A-B-CI,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4050   MEMSCAR_Q_R[]                     'WCTRL/MEMSCAR_WB,DQ1/Q_WX,RSRC/@1,ROT/ZERO,ALU/OR,MUX/R.S''
;4051   MEMSCAR_R[]_0                     'WCTRL/MEMSCAR_WB,RSRC/@1,SPW/RLONG,ROT/ZERO,ALPCTL/WX_S''
;4052   MEMSCAR_ZLIT24[]                  'WCTRL/MEMSCAR_WB,ALPCTL/WX_S,ROT/ZLIT24,LIT/LITRL,LITRL/@1''
;4053
;4054   MEMSCR_-1                         'WCTRL/MEMSCR_WB,ROT/MINUS1,ALPCTL/WX_S''
;4055   MEMSCR_0                          'WCTRL/MEMSCR_WB,ROT/ZERO,ALPCTL/WX_S''
;4056   MEMSCR_0_M[]_ZLIT0[]              'WCTRL/MEMSCR_WB,MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4057   MEMSCR_M[]                        'WCTRL/MEMSCR_WB,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZERO''
;4058   MEMSCR_M[].ANDNOT.(R[].RL.24)     'WCTRL/MEMSCR_WB,ALU/ANDNOT,MUX/M.S,MSRC/@1,ROT/RR.RR.SIZ,RSRC/@2,VSIZE/1,DTYPE/BYTE''
;4059   MEMSCR_M[].RL.24                  'WCTRL/MEMSCR_WB,ALPCTL/WX_S,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE''
;4060   MEMSCR_M[]_ZEXT(MB)               'WCTRL/MEMSCR_WB,MSRC/@1,SPW/MLONG,ROT/ZERO,MUX/XM.S,ALUXM/ZERO,ALU/OR''
;4061   MEMSCR_RTEMP7_M[]-ZLIT0[]         'WCTRL/MEMSCR_WB,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/A-B-CI,SPW/RLONG''
;4062   MEMSCR_R[]                        'WCTRL/MEMSCR_WB,RSRC/@1,MUX/R.S,ROT/ZERO,ALU/OR''
;4063   MEMSCR_R[]_M[].RL.24              'WCTRL/MEMSCR_WB,RSRC/@1,SPW/RLONG,MSRC/@2 ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE,ALPCTL/WX_S''
;4064   MEMSCR_ZLIT24[]                   'WCTRL/MEMSCR_WB,ALPCTL/WX_S,ROT/ZLIT24,LIT/LITRL,LITRL/@1''
;4065
;4066   MTEMP0_(MDR.RR.P).AND.R[]         ''SPW/MLONG,MSRC/MDR,RSRC/@1,ROT/RR.MM.P,ALU/AND,MUX/R.S''
;4067   MTEMP0_D_0_Q_SEXT(MDR)            ''SPW/MLONG,MSRC/MDR,RSRC/ZERO,ALPCTL/WX_D_R.Q_XM,ALUXM/SIGN''
;4068   MTEMP0_D_MDR-ZLIT0[]              ''SPW/MLONG,MSRC/MDR,ROT/ZLIT0,LIT/LITRL,LITRL/@1,MUX/M.S,ALU/A-B-CI,DQ1/D_WX''
;4069   MTEMP0_MDR+1                      ''SPW/MLONG,MSRC/MDR,ROT/MINUS1,MUX/M.S,ALU/A-B-CI''
;4070   MTEMP0_MDR+Q                      ''SPW/MLONG,MSRC/MDR,MUX/M.Q1,ALU/A+B+CI''
;4071   MTEMP0_MDR+R[]                    ''SPW/MLONG,MSRC/MDR,RSRC/@1,MUX/M.R1,ALU/A+B+CI''
;4072   MTEMP0_MDR+R[]+ALKC               ''SPW/MLONG,MSRC/MDR,RSRC/@1,MUX/M.R1,ALU/A+B+CI,ALUCI/ALKC''
;4073   MTEMP0_MDR-ZLIT0[]                ''SPW/MLONG,MSRC/MDR,ROT/ZLIT0,LIT/LITRL,LITRL/@1,MUX/M.S,ALU/A-B-CI''
;4074   MTEMP0_MDR.AND.OLIT8[]            ''SPW/MLONG,MSRC/MDR,LIT/LITRL,LITRL/@1,ROT/OLIT8,MUX/M.S,ALU/AND''
```

```
;4075   MTEMPO_MDR.AND.R[]                 "SPW/MLONG,MSRC/MDR,RSRC/@1,ALU/AND,MUX/M.R1"
;4076   MTEMPO_MDR.AND.ZLIT0[]             "SPW/MLONG,MSRC/MDR,LIT/LITRL,LITRL/@1,ROT/ZLIT0,MUX/M.S,ALU/AND"
;4077   MTEMPO_MDR.NOTAND.R[]              "SPW/MLONG,MSRC/MDR,RSRC/@1,MUX/M.R1,ALU/NOTAND"
;4078   MTEMPO_MDR.OR.R[]                  "SPW/MLONG,MSRC/MDR,RSRC/@1,MUX/M.R1,ALU/OR"
;4079   MTEMPO_MDR.OR.ZLIT0[]              "SPW/MLONG,MSRC/MDR,LIT/LITRL,LITRL/@1,ROT/ZLIT0,MUX/M.S,ALU/OR"
;4080   MTEMPO_MDR.OR.ZLIT16[]             "SPW/MLONG,MSRC/MDR,LIT/LITRL,LITRL/@1,ROT/ZLIT16,MUX/M.S,ALU/OR"
;4081   MTEMPO_MDR.OR.ZLIT8[]              "SPW/MLONG,MSRC/MDR,LIT/LITRL,LITRL/@1,ROT/ZLIT8,MUX/M.S,ALU/OR"
;4082   MTEMPO_MDR (MDR+ZLIT0[]).SL.1      "SPW/MLONG,MSRC/MDR,WCTRL/MDR_WB,ALU/A+B+CI.SL,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@1"
;4083   MTEMPO_PC-R[]                      "SPW/MLONG,MSRC/PC,RSRC/@1,MUX/M.R1,ALU/A-B-CI"
;4084   MTEMPO_PC.ANDNOT.R[]               "SPW/MLONG,MSRC/PC,RSRC/@1,MUX/M.R1,ALU/ANDNOT"
;4085   MTEMPO_PC.OR.R[]                   "SPW/MLONG,MSRC/PC,RSRC/@1,MUX/M.R1,ALU/OR"
;4086   MTEMPO_R[] Q_MDR                   "SPW/MLONG,MSRC/MDR,RSRC/@1,ALPCTL/WX_R.Q_M"
;4087   MTEMPO_R[]-PCBACK                  "SPW/MLONG,MSRC/PCBACK,ALU/B-A-CI,MUX/M.R1,RSRC/@1"
;4088   MTEMPO_R[].RR.24 Q_PCBACK          "SPW/MLONG,RSRC/@1,ROT/RR.RR.SIZ,MSRC/PCBACK,ALPCTL/WX_S.Q_XM,VSIZE/1,DTYPE/LONG"
;4089   MTEMPO_RBSP                        "MSRC/WB_RBSP,SPW/MLONG"
;4090   MTEMPO_SEXT(MDR).XOR.Q             "SPW/MLONG,MSRC/MDR,MUX/XM.Q,ALUXM/SIGN,ALU/XOR"
;4091   MTEMPO_SEXT(XB)+R[] PC_PC+2        "SPW/MLONG,MSRC/XB.PC_PC+1,RSRC/@1,MUX/XM.R,ALUXM/SIGN,ALU/A+B+CI,
;4092                                              ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/WORD"
;4093   MTEMPO_SEXT(XB)+R[] PC_PC+4        "SPW/MLONG,MSRC/XB.PC_PC+1,RSRC/@1,MUX/XM.R,ALUXM/SIGN,ALU/A+B+CI,
;4094                                              ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/LONG"
;4095   MTEMPO_SEXT(XB)+R[] PC_PC+1        "SPW/MLONG,MSRC/XB.PC_PC+1,RSRC/@1,MUX/XM.R,ALUXM/SIGN,ALU/A+B+CI"
;4096   MTEMPO_UNPACK(MDR R[])             "SPW/MLONG,MSRC/MDR,ROT/GETFPF,RSRC/@1,ALPCTL/WX_S"
;4097   MTEMPO_VA+ZLIT0[]                  "SPW/MLONG,MSRC/VA,LIT/LITRL,LITRL/@1,ROT/ZLIT0,MUX/M.S,ALU/A+B+CI"
;4098   MTEMPO_VA-R[]                      "SPW/MLONG,ALU/A-B-CI,MUX/M.R1,MSRC/VA,RSRC/@1"
;4099   MTEMPO_XB PC_PC+4                  "SPW/MLONG,ALU/OR,MUX/M.S,MSRC/XB.PC_PC+1,ROT/ZERO,ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/LONG"
;4100   MTEMPO_ZEXT(R[])                   "MSRC/@1,SPW/MLONG,ROT/ZERO,MUX/XM.S,ALU/OR,ALUXM/ZERO"
;4101   MTEMPO_ZEXT(XB) PC_PC+1            "SPW/MLONG,ALU/OR,MUX/XM.S,MSRC/XB.PC_PC+1,ROT/ZERO,ISTRM/ISIZE_DSIZE,
;4102                                              VSIZE/1,DTYPE/BYTE,ALUXM/ZERO"
;4103
;4104   M[]_(MB Q).RL.1                    'MSRC/@1,SPW/MLONG,RSRC/ZERO,ALU/A+B+CI.SL,MUX/M.R2,ALUSHF/ROT,ALUCI/ZERO,DQ2/SQL'
;4105   M[]_(MB Q).SL.1                    'MSRC/@1,SPW/MLONG,RSRC/ZERO,ALU/A+B+CI.SL,MUX/M.R2,ALUSHF/SHF,ALUCI/ZERO,DQ2/SQL'
;4106   M[]_(MB R[]).RR.4                  'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.MR.4,ALPCTL/WX_S'
;4107   M[]_(MB R[]).RR.9                  'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.MR.9,ALPCTL/WX_S'
;4108   M[]_(MB R[]).RR.P                  'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.MR.P,ALPCTL/WX_S'
;4109   M[]_(MB R[]).RR.PS                 'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.MR.PS,ALPCTL/WX_S'
;4110   M[]_(MB R[]).RR.S                  'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.MR.S,ALPCTL/WX_S'
;4111   M[]_(MB R[]).XZ                    'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/XZ.MR,ALPCTL/WX_S'
;4112   M[]_(MB+ALKC).BCD                  'MSRC/@1,SPW/MLONG,ALU/A+B+CI.BCD,MUX/M.R1,RSRC/ZERO,ALUCI/ALKC,VSIZE/1,DTYPE/LONG'
;4113   M[]_(MB+R[]).BCD                   'MSRC/@1,SPW/MLONG,RSRC/@2,ALU/A+B+CI.BCD,MUX/M.R1,VSIZE/1,DTYPE/LONG'
;4114   M[]_(MB+R[]).RL.1                  'MSRC/@1,SPW/MLONG,ALU/A+B+CI.SL,MUX/M.R1,ALUSHF/ROT,RSRC/@2'
;4115   M[]_(MB+R[]).RR.1                  'MSRC/@1,SPW/MLONG,ALU/A+B+CI.SR,MUX/M.R1,ALUSHF/ROT,RSRC/@2'
;4116   M[]_(MB+R[]).SL.1                  'MSRC/@1,SPW/MLONG,RSRC/@2,ALU/A+B+CI.SL,MUX/M.R1,ALUSHF/ZERO,ALUCI/ZERO'
;4117   M[]_(MB+R[]).SR.1                  'MSRC/@1,SPW/MLONG,RSRC/@2,ALU/A+B+CI.SR,MUX/M.R1,ALUSHF/ZERO,ALUCI/ZERO'
;4118   M[]_(MB+R[]+ALKC).BCD              'MSRC/@1,SPW/MLONG,RSRC/@2,ALU/A+B+CI.BCD,MUX/M.R1,ALUCI/ALKC,VSIZE/1,DTYPE/LONG'
;4119   M[]_(MB+R[]+ALKC).SL.1 ALU<0>_0    'MSRC/@1,SPW/MLONG,ALU/A+B+CI.SL,MUX/M.R1,ALUSHF/ZERO,RSRC/@2,ALUCI/ALKC'
;4120   M[]_(MB+R[]+ALKC).SL.1 ALU<0>_1    'MSRC/@1,SPW/MLONG,ALU/A+B+CI.SL,MUX/M.R1,ALUSHF/ONE,RSRC/@2,ALUCI/ALKC'
;4121   M[]_(MB+ZLIT24[]).BCD              'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT24,ALU/A+B+CI.BCD,MUX/M.S,VSIZE/1,DTYPE/LONG'
;4122   M[]_(MB-R[]).BCD                   'MSRC/@1,RSRC/@2,MUX/M.R1,VSIZE/1,DTYPE/LONG,ALU/A-B-CI.BCD,SPW/MLONG'
;4123   M[]_(MB-R[]).SL.1                  'MSRC/@1,SPW/MLONG,RSRC/@2,ALU/A-B-CI.SL,MUX/M.R1,ALUSHF/ZERO,ALUCI/ZERO'
;4124   M[]_(MB-R[]-ALKC).BCD              'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,VSIZE/1,DTYPE/LONG,ALU/A-B-CI.BCD,ALUCI/ALKC'
;4125   M[]_(MB-R[]-ALKC).SL.1 ALU<0>_0    'MSRC/@1,SPW/MLONG,ALU/A-B-CI.SL,MUX/M.R1,ALUSHF/ZERO,RSRC/@2,ALUCI/ALKC'
;4126   M[]_(MB-R[]-ALKC).SL.1 ALU<0>_1    'MSRC/@1,SPW/MLONG,ALU/A-B-CI.SL,MUX/M.R1,ALUSHF/ONE,RSRC/@2,ALUCI/ALKC'
;4127   M[]_(MB.RR.4).NOT                  'MSRC/@1,SPW/MLONG,ALPCTL/WX_.NOT.S,ROT/RR.MR.4,RSRC/@1'
;4128   M[]_(R[] MB).RL.4                  'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/RL.RM.4,RSRC/@2'
;4129   M[]_(R[] MB).RL.P                  'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/RL.RM.P,RSRC/@2'
```

```
;4130   M[]_(R[] MB).RL.PS                'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RL.RM.PS,ALPCTL/WX_S''
;4131   M[]_(R[]+PL).SR.1                 'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/PL,ALU/A+B+CI.SR,MUX/R.S''
;4132   M[]_(R[].ASL.1)-MB                'MSRC/@1,SPW/MLONG,ALU/B-A-CI,MUX/M.S,ROT/ASL.R.SIZ,VSIZE/1,DTYPE/WORD,RSRC/@2''
;4133   M[]_(R[].ASL.SIZE)                'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/ASL.R.SIZ,ALPCTL/WX_S''
;4134   M[]_-1                            'MSRC/@1,SPW/MLONG,ROT/MINUS1,ALPCTL/WX_S''
;4135   M[]_-MB                           'MSRC/@1,SPW/MLONG,ALUCI/ZERO,ALU/B-A-CI,RSRC/ZERO,MUX/M.R1''
;4136   M[]_-R[]                          'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/ZERO,MUX/R.S,ALU/B-A-CI''
;4137   M[]_0                             'MSRC/@1,SPW/MLONG,ROT/ZERO,ALPCTL/WX_S''
;4138   M[]_0+ZLIT0[]                     'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/Z.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4139   M[]_ASTLVL                        'MSRC/@1,SPW/MLONG,WCTRL/ASTLVL''
;4140   M[]_ATCR                          'MSRC/@1,SPW/MLONG,CCMISC/WB_ATCR.CCBR_SIGND''
;4141   M[]_CC_ZLIT0[]                    'MSRC/@1,SPW/MLONG,CCPSL/CC_WB.CCBR_ALUS,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4142   M[]_CONREGS                       'MSRC/@1,SPW/MLONG,WCTRL/CONREAD''
;4143   M[]_CONX(1)                       'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4144   M[]_CONX(2)                       'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD''
;4145   M[]_CONX(4)                       'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG''
;4146   M[]_CONX.SIZ                      'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/CONX.SIZ''
;4147   M[]_CVTNP(MB)                     'MSRC/@1,SPW/MLONG,ROT/CVTNP,ALU/A+B+CI.BCD,MUX/R.S,RSRC/ZERO''
;4148   M[]_D                             'MSRC/@1,SPW/MLONG,MUX/D.S,ROT/ZERO,ALU/OR''
;4149   M[]_D+(R[].RR.16)                 'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/D.S,ROT/RR.RR.SIZ,DTYPE/WORD,VSIZE/1,RSRC/@2''
;4150   M[]_D+ALKC                        'MSRC/@1,SPW/MLONG,MUX/D.R1,ALU/A+B+CI,RSRC/ZERO,ALUCI/ALKC''
;4151   M[]_D+R[]                         'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/D.R1,ALU/A+B+CI''
;4152   M[]_D+R[]+ALKC                    'MSRC/@1,SPW/MLONG,RSRC/@2,ALU/A+B+CI,MUX/D.R1,ALUCI/ALKC''
;4153   M[]_D+ZLIT12[]                    'MSRC/@1,SPW/MLONG,MUX/D.S,ROT/ZLIT12,LIT/LITRL,LITRL/@2,ALU/A+B+CI''
;4154   M[]_D+ZLIT8[]                     'MSRC/@1,SPW/MLONG,MUX/D.S,ROT/ZLIT8,LIT/LITRL,LITRL/@2,ALU/A+B+CI''
;4155   M[]_D-ZLIT0[]                     'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/D.S,ALU/A-B-CI''
;4156   M[]_D-ZLIT12[]                    'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT12,MUX/D.S,ALU/A-B-CI''
;4157   M[]_D-ZLIT8[]                     'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT8,MUX/D.S,ALU/A-B-CI''
;4158   M[]_D.AND.ZLIT0[]                 'MSRC/@1,SPW/MLONG,ALU/AND,MUX/D.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4159   M[]_D.AND.ZLIT8[]                 'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT8,MUX/D.S,ALU/AND''
;4160   M[]_D.ANDNOT.ZLIT0[]              'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/D.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4161   M[]_D.OR.(MB.RR.16)               'MSRC/@1,SPW/MLONG,ALU/OR,MUX/D.S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD''
;4162   M[]_D.OR.PACK(MB R[])             'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/FPACK,ALU/OR,MUX/D.S''
;4163   M[]_D.OR.ZLIT8[]                  'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT8,MUX/D.S,ALU/OR''
;4164   M[]_D_(MB R[]).RR.P               'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.MR.P,ALPCTL/WX_D_S''
;4165   M[]_D_(MB+Q).SL.1 Q<0>_1          'MSRC/@1,SPW/MLONG,DQ2/SQL.D_WX,ALU/A+B+CI.SL,MUX/M.Q2,ALUSHF/ALU0.Q1''
;4166   M[]_D_D.ANDNOT.ZLIT0[]            'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/D.S,ALU/ANDNOT,DQ1/D_WX''
;4167   M[]_D_SEXT(MB)                    'MSRC/@1,SPW/MLONG,DQ1/D_WX,MUX/XM.R,ALU/OR,RSRC/ZERO,ALUXM/SIGN''
;4168   M[]_D_Q                           'MSRC/@1,SPW/MLONG,RSRC/ZERO,MUX/R.Q,ALU/OR,DQ1/D_WX''
;4169   M[]_D_Q Q_MB                      'MSRC/@1,SPW/MLONG,ALPCTL/WX_D_Q.Q_M''
;4170   M[]_D_Q_0                         'MSRC/@1,SPW/MLONG,ALPCTL/WX_D_Q_S,ROT/ZERO''
;4171   M[]_D_Q_D+Q+1                     'MSRC/@1,SPW/MLONG,DQ1/Q_D_WX,ALU/A+B+CI,MUX/D.Q1,ALUCI/ONE''
;4172   M[]_D_REM                         'MSRC/@1,SPW/MLONG,RSRC/ZERO,ALPCTL/REM,ROT/0''
;4173   M[]_D_R[]                         'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/ZERO,MUX/R.S,ALU/OR,DQ1/D_WX''
;4174   M[]_D_ZLIT0[]                     'MSRC/@1,SPW/MLONG,ALPCTL/WX_D_S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4175   M[]_D_ZLIT12[]                    'MSRC/@1,SPW/MLONG,ALPCTL/WX_D_S,ROT/ZLIT12,LIT/LITRL,LITRL/@2''
;4176   M[]_D_ZLIT16[]                    'MSRC/@1,SPW/MLONG,DQ1/D_WX,ROT/ZLIT16,LIT/LITRL,LITRL/@2,MUX/Z.S,ALU/A+B+CI''
;4177   M[]_FLAGS                         'MSRC/@1,SPW/MLONG,WCTRL/CM.TP.FPD.FLAGS''
;4178   M[]_FPTCR                         'MSRC/@1,SPW/MLONG,WCTRL/FPTCR''
;4179   M[]_MB+(R[].ASL.1)                'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/M.S,ROT/ASL.R.SIZ,VSIZE/1,DTYPE/WORD,RSRC/@2''
;4180   M[]_MB+(R[].ASL.2)                'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/ASL.R.SIZ,VSIZE/1,DTYPE/LONG,ALU/A+B+CI,MUX/M.S''
;4181   M[]_MB+(R[].RR.24)                'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/M.S,ROT/RR.RR.SIZ,RSRC/@2,VSIZE/1,DTYPE/LONG''
;4182   M[]_MB+(R[].SL.2)                 'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/M.S,ROT/ASL.R.SIZ,VSIZE/1,DTYPE/LONG,RSRC/@2''
;4183   M[]_MB+1                          'MSRC/@1,SPW/MLONG,ALU/A+B+CI,ALUCI/ONE,MUX/M.R1,MSRC/@1,RSRC/ZERO''
;4184   M[]_MB+PL                         'MSRC/@1,SPW/MLONG,MUX/M.S,ALU/A+B+CI,ROT/PL''
```

```
;4185   M[]_MB+Q                          'MSRC/@1,SPW/MLONG,MUX/M.Q1,ALU/A+B+CI,ALUCI/ZERO''
;4186   M[]_MB+R[]                        'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,ALU/A+B+CI,ALUCI/ZERO''
;4187   M[]_MB+R[]+ALKC                   'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,ALU/A+B+CI,ALUCI/ALKC''
;4188   M[]_MB+ZLIT0[]                    'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/A+B+CI''
;4189   M[]_MB+ZLIT12[]                   'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT12,MUX/M.S,ALU/A+B+CI''
;4190   M[]_MB+ZLIT16[]                   'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/M.S,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4191   M[]_MB+ZLIT24[]                   'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/M.S,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;4192   M[]_MB+ZLIT8[]                    'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/M.S,ROT/ZLIT8,LIT/LITRL,LITRL/@2''
;4193   M[]_MB-1                          'MSRC/@1,SPW/MLONG,ALU/A-B-CI,ALUCI/ONE,MUX/M.R1,RSRC/ZERO''
;4194   M[]_MB-CONX(1)                    'MSRC/@1,SPW/MLONG,ALU/A-B-CI,MUX/M.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4195   M[]_MB-CONX(2)                    'MSRC/@1,SPW/MLONG,ALU/A-B-CI,MUX/M.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD''
;4196   M[]_MB-R[]                        'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,ALU/A-B-CI''
;4197   M[]_MB-R[]-ALKC                   'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,ALU/A-B-CI,ALUCI/ALKC''
;4198   M[]_MB-ZLIT0[]                    'MSRC/@1,SPW/MLONG,ALU/A-B-CI,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4199   M[]_MB-ZLIT12[]                   'MSRC/@1,SPW/MLONG,MUX/M.S,ALU/A-B-CI,ROT/ZLIT12,LIT/LITRL,LITRL/@2''
;4200   M[]_MB-ZLIT16[]                   'MSRC/@1,SPW/MLONG,ALU/A-B-CI,MUX/M.S,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4201   M[]_MB-ZLIT8[]                    'MSRC/@1,SPW/MLONG,ALU/A-B-CI,MUX/M.S,ROT/ZLIT8,LIT/LITRL,LITRL/@2''
;4202   M[]_MB.AND.CONX(1)                'MSRC/@1,SPW/MLONG,ALU/AND,MUX/M.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4203   M[]_MB.AND.OLIT0[]                'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/OLIT0,MUX/M.S,ALU/AND''
;4204   M[]_MB.AND.OLIT16[]               'MSRC/@1,SPW/MLONG,ALU/AND,MUX/M.S,ROT/OLIT16,LIT/LITRL,LITRL/@2''
;4205   M[]_MB.AND.OLIT24[]               'MSRC/@1,SPW/MLONG,ALU/AND,MUX/M.S,ROT/OLIT24,LIT/LITRL,LITRL/@2''
;4206   M[]_MB.AND.R[]                    'MSRC/@1,SPW/MLONG,ALU/AND,MUX/M.R1,RSRC/@2''
;4207   M[]_MB.AND.ZLIT0[]                'MSRC/@1,SPW/MLONG,ALU/AND,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4208   M[]_MB.AND.ZLIT16[]               'MSRC/@1,SPW/MLONG,ALU/AND,MUX/M.S,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4209   M[]_MB.AND.ZLIT24[]               'MSRC/@1,SPW/MLONG,ALU/AND,MUX/M.S,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;4210   M[]_MB.AND.ZLIT8[]                'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT8,MUX/M.S,ALU/AND''
;4211   M[]_MB.ANDNOT.CONX(1)             'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/CONX.SIZ,DTYPE/BYTE,VSIZE/1''
;4212   M[]_MB.ANDNOT.CONX(2)             'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/CONX.SIZ,DTYPE/WORD,VSIZE/1''
;4213   M[]_MB.ANDNOT.CONX(4)             'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/CONX.SIZ,DTYPE/LONG,VSIZE/1''
;4214   M[]_MB.ANDNOT.OLIT16[]            'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/OLIT16,LIT/LITRL,LITRL/@2''
;4215   M[]_MB.ANDNOT.R[]                 'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,ALU/ANDNOT''
;4216   M[]_MB.ANDNOT.ZLIT0[]             'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4217   M[]_MB.ANDNOT.ZLIT12[]            'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT12,MUX/M.S,ALU/ANDNOT''
;4218   M[]_MB.ANDNOT.ZLIT16[]            'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4219   M[]_MB.ANDNOT.ZLIT24[]            'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;4220   M[]_MB.ANDNOT.ZLIT28[]            'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/ZLIT28,LIT/LITRL,LITRL/@2''
;4221   M[]_MB.ANDNOT.ZLIT8[]             'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/ZLIT8,LIT/LITRL,LITRL/@2''
;4222   M[]_MB.ANDNOT.ZLITPL[]            'MSRC/@1,SPW/MLONG,ALU/ANDNOT,MUX/M.S,ROT/ZLITPL,LIT/LITRL,LITRL/@2''
;4223   M[]_MB.ASR.P                      'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/ASR.M.P''
;4224   M[]_MB.BCDSWP                     'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/BCDSWP''
;4225   M[]_MB.CLR1B                      'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/CLR1BM''
;4226   M[]_MB.CLR2B                      'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/CLR2BM''
;4227   M[]_MB.OR.(MB.RL.16)              'MSRC/@1,SPW/MLONG,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,MUX/M.S,ALU/OR''
;4228   M[]_MB.OR.(MB.RL.8)               'MSRC/@1,SPW/MLONG,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/LONG,MUX/M.S,ALU/OR''
;4229   M[]_MB.OR.(R[].RR.16)             'MSRC/@1,SPW/MLONG,RSRC/@2,ALU/OR,MUX/M.S,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/WORD''
;4230   M[]_MB.OR.(R[].RR.24)             'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/RR.RR.SIZ,RSRC/@2,VSIZE/1,DTYPE/LONG''
;4231   M[]_MB.OR.(R[].RR.8)              'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/RR.RR.SIZ,RSRC/@2,DTYPE/BYTE,VSIZE/1''
;4232   M[]_MB.OR.(R[].RL.8)              'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/RR.RR.SIZ,RSRC/@2,VSIZE/1,DTYPE/LONG''
;4233   M[]_MB.OR.CONX(1)                 'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/CONX.SIZ,DTYPE/BYTE,VSIZE/1''
;4234   M[]_MB.OR.CONX(2)                 'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/CONX.SIZ,DTYPE/WORD,VSIZE/1''
;4235   M[]_MB.OR.CONX(4)                 'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/CONX.SIZ,DTYPE/LONG,VSIZE/1''
;4236   M[]_MB.OR.R[]                     'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,ALU/OR''
;4237   M[]_MB.OR.ZLIT0[]                 'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4238   M[]_MB.OR.ZLIT12[]                'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/ZLIT12,LIT/LITRL,LITRL/@2''
;4239   M[]_MB.OR.ZLIT16[]                'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
```

```
;4240   M[]_MB.OR.ZLIT24[]          'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;4241   M[]_MB.OR.ZLIT28[]          'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT28,MUX/M.S,ALU/OR''
;4242   M[]_MB.OR.ZLIT8[]           'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/ZLIT8,LIT/LITRL,LITRL/@2''
;4243   M[]_MB.OR.ZLITPL[]          'MSRC/@1,SPW/MLONG,ALU/OR,MUX/M.S,ROT/ZLITPL,LIT/LITRL,LITRL/@2''
;4244   M[]_MB.RL.1                 'MSRC/@1,SPW/MLONG,RSRC/ZERO,ALU/A+B+CI.SL,MUX/M.R1,ALUCI/ZERO,ALUSHF/ROT''
;4245   M[]_MB.RL.24                'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE''
;4246   M[]_MB.RL.4                 'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/RL.RM.4,RSRC/@1''
;4247   M[]_MB.RL.8                 'MSRC/@1,SPW/MLONG,ROT/RR.MM.SIZ,ALPCTL/WX_S,VSIZE/1,DTYPE/LONG''
;4248   M[]_MB.RR.16                'MSRC/@1,SPW/MLONG,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,ALPCTL/WX_S''
;4249   M[]_MB.RR.24                'MSRC/@1,SPW/MLONG,ROT/RR.MM.SIZ,ALPCTL/WX_S,VSIZE/1,DTYPE/LONG''
;4250   M[]_MB.RR.4                 'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/RR.MR.4,RSRC/@1''
;4251   M[]_MB.RR.8                 'MSRC/@1,SPW/MLONG,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE,ALPCTL/WX_S''
;4252   M[]_MB.RR.P                 'MSRC/@1,SPW/MLONG,ROT/RR.MM.P,ALPCTL/WX_S''
;4253   M[]_MB.SR.1                 'MSRC/@1,SPW/MLONG,RSRC/ZERO,ALU/A+B+CI.SR,MUX/M.R1,ALUCI/ZERO,ALUSHF/ZERO''
;4254   M[]_MB.XOR.ZLIT12[]         'MSRC/@1,SPW/MLONG,ROT/ZLIT12,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/XOR''
;4255   M[]_MEMSCR                  'MSRC/@1,SPW/MLONG,WCTRL/MEMSCR''
;4256   M[]_OLIT0[]                 'MSRC/@1,SPW/MLONG,ROT/OLIT0,LIT/LITRL,LITRL/@2,ALPCTL/WX_S''
;4257   M[]_OLIT16[]                'MSRC/@1,SPW/MLONG,ROT/OLIT16,LIT/LITRL,LITRL/@2,ALPCTL/WX_S''
;4258   M[]_OLIT24[]                'MSRC/@1,SPW/MLONG,ROT/OLIT24,LIT/LITRL,LITRL/@2,ALPCTL/WX_S''
;4259   M[]_OLIT8[]                 'MSRC/@1,SPW/MLONG,ROT/OLIT8,LIT/LITRL,LITRL/@2,ALPCTL/WX_S''
;4260   M[]_PACK(MB R[])            'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/FPACK,ALPCTL/WX_S''
;4261   M[]_PL                      'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/PL''
;4262   M[]_PL_Q                    'MSRC/@1,SPW/MLONG,ROT/SL.PL_WB,RSRC/ZERO,MUX/R.Q,ALU/OR''
;4263   M[]_PSL                     'MSRC/@1,SPW/MLONG,CCPSL/WB_PSL.CCBR_SIGND''
;4264   M[]_Q                       'MSRC/@1,SPW/MLONG,RSRC/ZERO,MUX/R.Q,ALU/OR''
;4265   M[]_Q Q_D                   'MSRC/@1,SPW/MLONG,ALPCTL/WX_Q.Q_D''
;4266   M[]_Q Q_MB PL_[]            'MSRC/@1,SPW/MLONG,ALPCTL/WX_Q.Q_M,ROT/OLIT0.PL_LIT,LIT/LITRL,LITRL/@2''
;4267   M[]_Q_D+R[]+ALKC            'MSRC/@1,SPW/MLONG,RSRC/@2,ALU/A+B+CI,MUX/D.R1,ALUCI/ALKC,DQ1/Q_WX''
;4268   M[]_Q_D_R[]                 'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/ZERO,MUX/R.S,ALU/OR,DQ1/Q_D_WX''
;4269   M[]_Q_FLAGS_R[]             'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/ZERO,MUX/R.S,ALU/OR,DQ1/Q_WX,WCTRL/FLAGS_WB''
;4270   M[]_Q_MB.ASR.P              'MSRC/@1,SPW/MLONG,ALPCTL/WX_Q_S,ROT/ASR.M.P''
;4271   M[]_Q_MB.XOR.R[]            'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,DQ1/Q_WX,ALU/XOR''
;4272   M[]_Q_R[]                   'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/R.S,ROT/ZERO,ALU/OR,DQ1/Q_WX''
;4273   M[]_Q_UNPACK(MB R[])        'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/GETFPF,ALPCTL/WX_Q_S''
;4274   M[]_Q_ZLIT0[]               'MSRC/@1,SPW/MLONG,ALPCTL/WX_Q_S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4275   M[]_R[]                     'MSRC/@1,SPW/MLONG,ALU/OR,MUX/R.S,ROT/ZERO,RSRC/@2''
;4276   M[]_R[] Q_MB                'MSRC/@1,SPW/MLONG,RSRC/@2,ALPCTL/WX_R.Q_M''
;4277   M[]_R[] Q_ZEXT(MB)          'MSRC/@1,SPW/MLONG,ALPCTL/WX_R.Q_XM,RSRC/@2,ALUXM/ZERO''
;4278   M[]_R[]+0                   'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/R.S,RSRC/@2,ROT/ZERO''
;4279   M[]_R[]+1                   'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/MINUS1,MUX/R.S,ALU/A-B-CI''
;4280   M[]_R[]+Q                   'MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/R.Q,RSRC/@2''
;4281   M[]_R[]-0                   'MSRC/@1,SPW/MLONG,ALU/A-B-CI,MUX/R.S,RSRC/@2,ROT/ZERO''
;4282   M[]_R[]-1                   'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/MINUS1,MUX/R.S,ALU/A+B+CI''
;4283   M[]_R[]-D-ALKC              'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/D.R1,ALUCI/ALKC,ALU/B-A-CI''
;4284   M[]_R[]-MB                  'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,ALU/B-A-CI,ALUCI/ZERO''
;4285   M[]_R[]-MB-ALKC             'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,ALU/B-A-CI,ALUCI/ALKC''
;4286   M[]_R[]-Q                   'MSRC/@1,SPW/MLONG,ALU/A-B-CI,MUX/R.Q,RSRC/@2''
;4287   M[]_R[].ASL.1               'MSRC/@1,SPW/MLONG,RSRC/@2,ALPCTL/WX_S,ROT/ASL.R.SIZ,VSIZE/1,DTYPE/WORD''
;4288   M[]_R[].ASL.P               'MSRC/@1,SPW/MLONG,RSRC/@2,ALPCTL/WX_S,ROT/ASL.R.P''
;4289   M[]_R[].NOT                 'MSRC/@1,SPW/MLONG,ALU/NOTAND,MUX/R.S,ROT/MINUS1,RSRC/@2''
;4290   M[]_R[].OR.((MB RB).RR.4)   'MSRC/@1,SPW/MLONG,ROT/RR.MR.4,RSRC/@2,MUX/R.S,ALU/OR''
;4291   M[]_R[].OR.((RB MB).RL.4)   'MSRC/@1,SPW/MLONG,ROT/RL.RM.4,RSRC/@2,MUX/R.S,ALU/OR''
;4292   M[]_R[].OR.D                'MSRC/@1,SPW/MLONG,RSRC/@2,ALU/OR,MUX/D.R1''
;4293   M[]_R[].RR.16               'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/WORD,RSRC/@2''
;4294   M[]_R[].RR.24               'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/LONG,ALPCTL/WX_S''
```

```
;4295   M[]_R[].RR.8                    'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/RR.RR.SIZ,RSRC/@2,VSIZE/1,DTYPE/BYTE''
;4296   M[]_R[].RR.PS                   'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.RR.PS,ALPCTL/WX_S''
;4297   M[]_R[].RR.SIZ                  'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/RR.RR.SIZ,ALPCTL/WX_S''
;4298   M[]_R[].SL.1                    'MSRC/@1,SPW/MLONG,ALU/A+B+CI.SL,MUX/R.S,RSRC/@2,ROT/ZERO''
;4299   M[]_R[].SR.1                    'MSRC/@1,SPW/MLONG,ALU/A+B+CI.SR,MUX/R.S,RSRC/@2,ROT/ZERO''
;4300   M[]_R[].XZ                      'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/XZ.RR,ALPCTL/WX_S''
;4301   M[]_SEXT(MB)                    'MSRC/@1,SPW/MLONG,RSRC/ZERO,ALUXM/SIGN,MUX/XM.R,ALU/OR''
;4302   M[]_SEXT(MB).XOR.Q              'MSRC/@1,SPW/MLONG,MUX/XM.Q,ALUXM/SIGN,ALU/XOR''
;4303   M[]_STEPC                       'MSRC/@1,SPW/MLONG,WCTRL/CM.TP.FPD.F5.STEPC''
;4304   M[]_TCSR.IICR                   'MSRC/@1,SPW/MLONG,WCTRL/TCSR.IICR''
;4305   M[]_TOYOS.TOYCN                 'MSRC/@1,SPW/MLONG,WCTRL/TODCLK''
;4306   M[]_TRAR_ZLIT16[]               'MSRC/@1,SPW/MLONG,WCTRL/LOADTRAR,ALPCTL/WX_S,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4307   M[]_TU58REGS                    'MSRC/@1,SPW/MLONG,WCTRL/TU58READ''
;4308   M[]_UNPACK(MB R[])              'MSRC/@1,SPW/MLONG,ROT/GETFPF,RSRC/@2,ALPCTL/WX_S''
;4309   M[]_VA_R[]-1                    'MSRC/@1,SPW/MLONG,RSRC/@2,ROT/MINUS1,MUX/R.S,ALU/A+B+CI,WCTRL/VA_WB''
;4310   M[]_WB                          ''SPW/MLONG,MSRC/@1''
;4311   M[]_ZEXT(MB)                    'MSRC/@1,SPW/MLONG,ROT/ZERO,ALU/OR,MUX/XM.S,ALUXM/ZERO''
;4312   M[]_ZEXT(MB)+R[]                'MSRC/@1,SPW/MLONG,RSRC/@2,MUX/XM.R,ALU/A+B+CI,ALUXM/ZERO''
;4313   M[]_ZEXT(MB)-R[]                'MSRC/@1,SPW/MLONG,ALU/A-B-CI,MUX/XM.R,RSRC/@2,ALUXM/ZERO''
;4314   M[]_ZEXT(MB).SR.1               'MSRC/@1,SPW/MLONG,ALU/A+B+CI.SR,MUX/XM.R,RSRC/ZERO,ALUSHF/ZERO''
;4315   M[]_ZLIT0[]                     'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4316   M[]_ZLIT0[].Q_0                 'MSRC/@1,SPW/MLONG,ALPCTL/WX_S.Q_0,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4317   M[]_ZLIT0[]-MB                  'MSRC/@1,SPW/MLONG,ALU/B-A-CI,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4318   M[]_ZLIT8[]-MB                  'MSRC/@1,SPW/MLONG,ALU/B-A-CI,MUX/M.S,ROT/ZLIT8,LIT/LITRL,LITRL/@2''
;4319   M[]_ZLIT12[]                    'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/ZLIT12,LIT/LITRL,LITRL/@2''
;4320   M[]_ZLIT16[]                    'MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4321   M[]_ZLIT24[]                    'MSRC/@1,SPW/MLONG,ROT/ZLIT24,LIT/LITRL,LITRL/@2,ALPCTL/WX_S''
;4322   M[]_ZLIT28[]                    'MSRC/@1,SPW/MLONG,ROT/ZLIT28,LIT/LITRL,LITRL/@2,ALPCTL/WX_S''
;4323   M[]_ZLIT8[]                     'MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT8,ALPCTL/WX_S''
;4324
;4325   PC_-M[]                         'WCTRL/PC_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/B-A-CI''
;4326   PC_D_D+OLIT0[]                  'WCTRL/PC_WB,LIT/LITRL,LITRL/@1,ROT/OLIT0,MUX/D.S,ALU/A+B+CI,DQ1/D_WX''
;4327   PC_D_D-1                        'WCTRL/PC_WB,MUX/D.R1,RSRC/ZERO,ALU/A-B-CI,ALUCI/ONE,DQ1/D_WX''
;4328   PC_D_D-ZLIT0[]                  'WCTRL/PC_WB,LIT/LITRL,LITRL/@1,ROT/ZLIT0,MUX/D.S,ALU/A-B-CI,DQ1/D_WX''
;4329   PC_D_M[]                        'WCTRL/PC_WB,MSRC/@1,ROT/ZERO,MUX/M.S,ALU/OR,DQ1/D_WX''
;4330   PC_D_M[]-1                      'WCTRL/PC_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/A-B-CI,ALUCI/ONE,DQ1/D_WX''
;4331   PC_D_M[]-ZLIT0[]                'WCTRL/PC_WB,MSRC/@1,LITRL/@2,LIT/LITRL,ROT/ZLIT0,MUX/M.S,ALU/A-B-CI,DQ1/D_WX''
;4332   PC_M[]                          'WCTRL/PC_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/OR''
;4333   PC_M[]+(R[].RR.16)              'WCTRL/PC_WB,ALU/A+B+CI,MUX/M.S,MSRC/@1,ROT/RR.RR.SIZ,RSRC/@2,VSIZE/1,DTYPE/WORD''
;4334   PC_M[]+PSLC                     'WCTRL/PC_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/A+B+CI,ALUCI/PSLC''
;4335   PC_M[]+Q                        'WCTRL/PC_WB,MSRC/@1,MUX/M.Q1,ALU/A+B+CI''
;4336   PC_M[]+R[]                      'WCTRL/PC_WB,MSRC/@1,RSRC/@2,ALU/A+B+CI,ALUCI/ZERO,MUX/M.R1''
;4337   PC_M[]+ZLIT0[]                  'WCTRL/PC_WB,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/A+B+CI''
;4338   PC_M[]+ZLIT4[]                  'WCTRL/PC_WB,MSRC/@1,ROT/ZLIT4,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/A+B+CI''
;4339   PC_M[]+ZLIT8[]                  'WCTRL/PC_WB,MSRC/@1,ROT/ZLIT8,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/A+B+CI''
;4340   PC_M[]-CONX(2)                  'WCTRL/PC_WB,ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD''
;4341   PC_M[]+CONX.SIZ                 'WCTRL/PC_WB,ALU/A+B+CI,MUX/M.S,MSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/IDEP''
;4342   PC_M[]-PSLC                     'WCTRL/PC_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/A-B-CI,ALUCI/PSLC''
;4343   PC_M[]-R[]                      'WCTRL/PC_WB,MSRC/@1,RSRC/@2,ALU/A-B-CI,ALUCI/ZERO,MUX/M.R1''
;4344   PC_M[]-ZLIT0[]                  'WCTRL/PC_WB,ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4345   PC_M[].ANDNOT.ZLIT0[]           'WCTRL/PC_WB,ALU/ANDNOT,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4346   PC_M[].NOTAND.R[]               'WCTRL/PC_WB,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/NOTAND''
;4347   PC_M[].OR.R[]                   'WCTRL/PC_WB,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/OR''
;4348   PC_M[].OR.ZLIT0[]               'WCTRL/PC_WB,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4349   PC_M[].RR.P                     'WCTRL/PC_WB,MSRC/@1,ROT/RR.MM.P,ALPCTL/WX_S''
```

```
;4350   PC_M[].SL.1                    'WCTRL/PC_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/A+B+CI.SL''
;4351   PC_M[].SR.1                    'WCTRL/PC_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/A+B+CI.SR''
;4352   PC_M[].XOR.Q                   'WCTRL/PC_WB,MSRC/@1,MUX/M.Q1,ALU/XOR''
;4353   PC_M[]_D                       'WCTRL/PC_WB,MSRC/@1,SPW/MLONG,ALU/OR,MUX/D.S,ROT/ZERO''
;4354   PC_M[]_R[]                     'WCTRL/PC_WB,MSRC/@1,RSRC/@2,ROT/ZERO,MUX/R.S,ALU/OR,SPW/MLONG''
;4355   PC_PC+?((D+ZLITO[]).SL.1)      'WCTRL/PC_PC+WB,LIT/LITRL,LITRL/@1,ROT/ZLITO,MUX/D.S,ALU/A+B+CI.SL''
;4356   PC_PC+(M[].SL.1)               'WCTRL/PC_PC+WB,MSRC/@1,ROT/MINUS1,MUX/M.S,ALU/AND.SL''
;4357   PC_PC+(SEXT(M[]).SL.1)         'WCTRL/PC_PC+WB,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/A+B+CI.SL''
;4358   PC_PC+1 MB_XB                  'MSRC/XB.PC_PC+I,ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/BYTE''
;4359   PC_PC+2 MB_XB                  'MSRC/XB.PC_PC+I,ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/WORD''
;4360   PC_PC+D                        'WCTRL/PC_PC+WB,MUX/D.R1,RSRC/ZERO,ALU/OR''
;4361   PC_PC+I MB_XB                  'MSRC/XB.PC_PC+I,ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/IDEP''
;4362   PC_PC+M[]                      'WCTRL/PC_PC+WB,MUX/M.R1,MSRC/@1,RSRC/ZERO,ALU/OR''
;4363   PC_PC+Q                        'WCTRL/PC_PC+WB,MUX/R.Q,RSRC/ZERO,ALU/OR''
;4364   PC_PC+SEXT(M[])                'WCTRL/PC_PC+WB,MSRC/@1,RSPC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/OR''
;4365   PC_PC+SEXT(XB)+1               'WCTRL/PC_PC+WB,MSRC/XB.PC_PC+I,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/A+B+CI,ALUCI/ONE,
;4366                                                  ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/BYTE''
;4367   PC_Q                           'WCTRL/PC_WB,RSRC/ZERO,MUX/R.Q,ALU/OR''
;4368   PC_Q-M[]                       'WCTRL/PC_WB,MSRC/@1,MUX/M.Q1,ALU/B-A-CI''
;4369   PC_Q-M[]-1                     'WCTRL/PC_WB,MSRC/@1,MUX/M.Q1,ALU/B-A-CI,ALUCI/ONE''
;4370   PC_Q_M[].RR.SIZ                'WCTRL/PC_WB,MSRC/@1,ROT/RR.MM.SIZ,ALPCTL/WX_Q_S''
;4371   PC_R[]                         'WCTRL/PC_WB,ALU/OR,MUX/R.S,RSRC/@1,ROT/ZERO''
;4372   PC_R[]+CONX(2)                 'WCTRL/PC_WB,ALU/A+B+CI,MUX/R.S,RSRC/@1,ROT/CONX.SIZ,DTYPE/WORD,VSIZE/1''
;4373   PC_R[]_M[]                     'WCTRL/PC_WB,SPW/RLONG,RSRC/@1,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/OR''
;4374   PC_R[]_M[]-1                   'WCTRL/PC_WB,MSRC/@2,RSRC/@1,ROT/MINUS1,MUX/M.S,ALU/A+B+CI,SPW/RLONG''
;4375   PC_R[]_M[]-CONX(2)             'WCTRL/PC_WB,RSRC/@1,SPW/RLONG,MSRC/@2,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD,MUX/M.S,ALU/A-B-CI''
;4376   PC_R[]_Q_Q_D                   'WCTRL/PC_WB,SPW/RLONG,RSRC/@1,ALPCTL/WX_Q.Q_D''
;4377   PC_R[]_RB+1                    'WCTRL/PC_WB,RSRC/@1,ROT/MINUS1,MUX/R.S,ALU/A-B-CI,SPW/RLONG''
;4378   PC_R[]_RB-1                    'WCTRL/PC_WB,RSRC/@1,ROT/MINUS1,MUX/R.S,ALU/A+B+CI,SPW/RLONG''
;4379   PC_SEXT(M[])                   'WCTRL/PC_WB,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/OR''
;4380   PC_ZEXT(M[])                   'WCTRL/PC_WB,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALUXM/ZERO,ALU/OR''
;4381   PC_ZEXT(M[])+Q                 'WCTRL/PC_WB,ALU/A+B+CI,MUX/XM.Q,MSRC/@1,ALUXM/ZERO''
;4382   PC_ZEXT(M[])+R[]               'WCTRL/PC_WB,ALU/A+B+CI,MUX/XM.R,MSRC/@1,RSRC/@2,ALUXM/ZERO''
;4383   PC_ZEXT(XB<7-0>)+R[]           'WCTRL/PC_WB,MSRC/XB.PC_PC+I,RSRC/@1,MUX/XM.R,ALUXM/ZERO,ALU/A+B+CI,
;4384                                                  ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/BYTE''
;4385   PC_ZLITO[]                     'WCTRL/PC_WB,ALPCTL/WX_S,ROT/ZLITO,LIT/LITRL,LITRL/@1''
;4386
;4387   PL<4-3>_WB                     ''ROT/OLITO.PL43_WB''
;4388   PL_-D                          ''ROT/SL.PL_WB,RSRC/ZERO,MUX/D.R1,ALU/B-A-CI''
;4389   PL_-M[]                        ''ROT/SL.PL_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/B-A-CI''
;4390   PL_D_M[]-R[]                   ''ROT/SL.PL_WB,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/A-B-CI,DQ1/D_WX''
;4391   PL_D_Q_0                       ''ROT/SL.PL_WB,DQ1/Q_D_WX,ALU/AND,MUX/R.S,RSRC/ZERO''
;4392   PL_MSS_M[]                     ''ROT/PL_MSS,MSRC/@1,RSRC/ZERO,ALU/OR,MUX/M.R1''
;4393   PL_M[]                         ''ROT/SL.PL_WB,ALU/OR,MUX/M.R1,MSRC/@1,RSRC/ZERO''
;4394   PL_M[]_0                       ''ROT/SL.PL_WB,MSRC/@1,SPW/MLONG,ALU/AND,MUX/R.S,RSRC/ZERO''
;4395   PL_M[]_D_Q_0                   ''ROT/SL.PL_WB,MSRC/@1,SPW/MLONG,DQ1/Q_D_WX,ALU/AND,MUX/R.S,RSRC/ZERO''
;4396   PL_M[]_MB+Q                    ''ROT/SL.PL_WB,MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/M.Q1''
;4397   PL_Q                           ''ROT/SL.PL_WB,MUX/R.Q,RSRC/ZERO,ALU/OR''
;4398   PL_Q-M[]                       ''ROT/SL.PL_WB,MSRC/@1,MUX/M.Q1,ALU/B-A-CI''
;4399   PL_Q_M[]-R[]                   ''ROT/SL.PL_WB,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/A-B-CI,DQ1/Q_WX''
;4400   PL_Q_R[]-M[]                   ''ROT/SL.PL_WB,MSRC/@2,RSRC/@1,MUX/M.R1,ALU/B-A-CI,DQ1/Q_WX''
;4401   PL_R[]-M[]                     ''ROT/SL.PL_WB,RSRC/@1,MSRC/@2,MUX/M.R1,ALU/B-A-CI''
;4402   PL_R[]_RB+Q                    ''ROT/SL.PL_WB,RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/R.Q''
;4403   PL_[]                          ''ROT/OLITO.PL_LIT,LIT/LITRL,LITRL/@1''
;4404
```

```
;4405   PME_0                          ''ROT/ZERO,ALPCTL/WX_S,WCTRL/PME''
;4406   PME_0 FPDOFFSET_3              ''WCTRL/PME,SPW/MLONG,MSRC/FPDOFFSET,ROT/ZLIT0,LIT/LITRL,LITRL/3,ALPCTL/WX_S''
;4407   PME_Q_M[].AND.ZLIT24[]         ''WCTRL/PME,DQ1/Q_WX,ALU/AND,MUX/M.S,MSRC/@1,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;4408   PME_Q.RR.1                     ''WCTRL/PME,ALU/A+B+CI.SR,MUX/R.Q,RSRC/ZERO,ALUSHF/ROT''
;4409
;4410   PSL(PREV_CURM ISCURM_R[])      ''WCTRL/PREV_CUR.ISCUR_WB,RSRC/@1,MUX/R.S,ROT/ZERO,ALU/OR''
;4411   PSL(PREV_CURM ISCURM_[])       ''WCTRL/PREV_CUR.ISCUR_WB,ALPCTL/WX_S,ROT/ZLIT24,LIT/LITRL,LITRL/@1''
;4412   PSL<CURM>_M[].RR.8             ''WCTRL/PREV_CUR.ISCUR_WB,ALPCTL/WX_S,ROT/RR.MM.SIZ,MSRC/@1,VSIZE/1,DTYPE/BYTE''
;4413   PSL<NZVC>_0                    ''CCPSL/CC_WB.CCBR_ALUS,ALPCTL/WX_S,ROT/ZERO''
;4414   PSL_M[]                        ''CCPSL/PSL_WB.CCBR_ALUS,ALU/OR,MUX/M.R1,RSRC/ZERO,MSRC/@1''
;4415   PSL_M[].ANDNOT.ZLIT0[]         ''CCPSL/PSL_WB.CCBR_ALUS,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/ANDNOT''
;4416   PSL_M[].ANDNOT.ZLIT24[]        ''CCPSL/PSL_WB.CCBR_ALUS,ALU/ANDNOT,MUX/M.S,MSRC/@1,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;4417   PSL_M[].OR.ZLIT0[]             ''CCPSL/PSL_WB.CCBR_ALUS,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/OR''
;4418   PSL_M[].OR.ZLIT24[]            ''CCPSL/PSL_WB.CCBR_ALUS,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;4419   PSL_M[]_MB<31-16>.0            ''CCPSL/PSL_WB.CCBR_ALUS,MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/CLR2BM''
;4420   PSL_M[]_ZLIT0[]                ''CCPSL/PSL_WB.CCBR_ALUS,MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4421   PSL_R[]                        ''CCPSL/PSL_WB.CCBR_ALUS,RSRC/@1,ROT/ZERO,MUX/R.S,ALU/XOR''
;4422   PSL_ZLIT16[]                   ''CCPSL/PSL_WB.CCBR_ALUS,ALPCTL/WX_S,ROT/ZLIT16,LIT/LITRL,LITRL/@1''
;4423
;4424   PSW_M[]                        ''CCPSL/PSW_WB.CCBR_ALUS,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/OR''
;4425   PSW_M[].AND.ZLIT0[]            ''CCPSL/PSW_WB.CCBR_ALUS,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/AND''
;4426   PSW_M[].NOTAND.R[]             ''CCPSL/PSW_WB.CCBR_ALUS,ALU/NOTAND,MUX/M.R1,MSRC/@1,RSRC/@2''
;4427   PSW_M[].OR.R[]                 ''CCPSL/PSW_WB.CCBR_ALUS,ALU/OR,MUX/M.R1,MSRC/@1,RSRC/@2''
;4428
;4429   Q_(M[] R[]).RR.P               ''ALPCTL/WX_Q_S,MSRC/@1,RSRC/@2,ROT/RR.MR.P''
;4430   Q_(R[] M[]).RL.4               ''ALPCTL/WX_Q_S,ROT/RL.RM.4,RSRC/@1,MSRC/@2''
;4431   Q_(R[] M[]).RL.P               ''ALPCTL/WX_Q_S,RSRC/@1,MSRC/@2,ROT/RL.RM.P''
;4432   Q_(R[]+Q).RR.1                 ''DQ1/Q_WX,ALU/A+B+CI.SR,MUX/R.Q,RSRC/@1,ALUSHF/ROT''
;4433   Q_(R[]+Q).SL.1                 ''DQ1/Q_WX,ALU/A+B+CI.SL,MUX/R.Q,RSRC/@1''
;4434   Q_-1                           ''ALPCTL/WX_Q_S,ROT/MINUS1''
;4435   Q_-SEXT(M[])                   ''DQ1/Q_WX,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALU/B-A-CI,ALUXM/SIGN''
;4436   Q_0                            ''ALPCTL/WX_Q_S,ROT/ZERO''
;4437   Q_D+Q                          ''DQ1/Q_WX,MUX/D.Q1,ALU/A+B+CI,ALUCI/ZERO''
;4438   Q_D.XOR.(M[].RL.P)             ''DQ1/Q_WX,ALU/XOR,MUX/D.S,ROT/RL.MM.P,MSRC/@1''
;4439   Q_MDR VA_MTEMP0_R[]            ''ALPCTL/WX_R.Q_M,RSRC/@1,MSRC/MDR,SPW/MLONG,WCTRL/VA_WB''
;4440   Q_M[]                          ''DQ1/Q_WX,MSRC/@1,ROT/ZERO,MUX/M.S,ALU/OR''
;4441   Q_M[] D_QOLD                   ''ALPCTL/WX_D_Q.Q_M,MSRC/@1''
;4442   Q_M[] D_R[]                    ''ALPCTL/WX_D_R.Q_M,MSRC/@1,RSRC/@2''
;4443   Q_M[] MDR_R[]                  ''ALPCTL/WX_R.Q_M,WCTRL/MDR_WB,MSRC/@1,RSRC/@2''
;4444   Q_M[] VA_ZLIT0[]               ''ALPCTL/WX_S.Q_XM,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2,WCTRL/VA_WB,VSIZE/1,DTYPE/LONG''
;4445   Q_M[] MDR_ZLIT0[]              ''ALPCTL/WX_S.Q_XM,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2,WCTRL/MDR_WB,VSIZE/1,DTYPE/LONG''
;4446   Q_M[] VA_R[]                   ''ALPCTL/WX_R.Q_M,WCTRL/VA_WB,MSRC/@1,RSRC/@2''
;4447   Q_M[] WB_R[]                   ''ALPCTL/WX_R.Q_M,MSRC/@1,RSRC/@2''
;4448   Q_M[]+R[]                      ''DQ1/Q_WX,ALU/A+B+CI,MUX/M.R1,MSRC/@1,RSRC/@2''
;4449   Q_M[]+R[]+ALKC                 ''DQ1/Q_WX,ALU/A+B+CI,MUX/M.R1,MSRC/@1,RSRC/@2,ALUCI/ALKC''
;4450   Q_M[]-Q                        ''DQ1/Q_WX,MSRC/@1,MUX/M.Q1,ALU/A-B-CI''
;4451   Q_M[]-R[]                      ''DQ1/Q_WX,MSRC/@1,RSRC/@2,ALU/A-B-CI,MUX/M.R1''
;4452   Q_M[]-ZLIT0[]                  ''DQ1/Q_WX,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/A-B-CI''
;4453   Q_M[]-ZLIT8[]                  ''DQ1/Q_WX,MSRC/@1,ROT/ZLIT8,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/A-B-CI''
;4454   Q_M[].AND.((MB R[]).RR.4)      ''DQ1/Q_WX,ALU/AND,MUX/M.S,MSRC/@1,ROT/RR.MR.4,RSRC/@2''
;4455   Q_M[].AND.(R[].RR.24)          ''DQ1/Q_WX,ALU/AND,MUX/M.S,MSRC/@1,ROT/RR.RR.SIZ,RSRC/@2,VSIZE/1,DTYPE/LONG''
;4456   Q_M[].AND.R[]                  ''DQ1/Q_WX,ALU/AND,MUX/M.R1,MSRC/@1,RSRC/@2''
;4457   Q_M[].AND.ZLIT0[]              ''DQ1/Q_WX,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/AND''
;4458   Q_M[].ANDNOT.ZLIT0[]           ''DQ1/Q_WX,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/ANDNOT''
;4459   Q_M[].ASL.P                    ''ALPCTL/WX_Q_S,MSRC/@1,ROT/ASL.M.P''
```

```
;4460   Q_M[].RL.P                        ''ALPCTL/WX_Q_S,MSRC/@1,ROT/RL.MM.P''
;4461   Q_M[].RR.16                       ''ALPCTL/WX_Q_S,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD''
;4462   Q_M[].SL.1                        ''DQ1/Q_WX,ALU/AND.SL,MUX/M.S,MSRC/@1,ROT/MINUS1''
;4463   Q_M[].XZ.MM                       ''ALPCTL/WX_Q_S,MSRC/@1,ROT/XZ.MM''
;4464   Q_M[]_R[]                         ''DQ1/Q_WX,MSRC/@1,SPW/MLONG,ALU/OR,MUX/R.S,ROT/ZERO,RSRC/@2''
;4465   Q_OLITO[]                         ''ROT/OLITO,LIT/LITRL,LITRL/@1,ALPCTL/WX_Q_S''
;4466   Q_Q+M[]                           ''DQ1/Q_WX,MSRC/@1,MUX/M.Q1,ALU/A+B+CI''
;4467   Q_Q+R[]                 ''DQ1/Q_WX,RSRC/@1,MUX/R.Q,ALU/B-A-CI''
;4468   Q_Q-M[]                           ''DQ1/Q_WX,MSRC/@1,MUX/M.Q1,ALU/B-A-CI''
;4469   Q_Q-R[]                           ''DQ1/Q_WX,RSRC/@1,MUX/R.Q,ALU/B-A-CI''
;4470   Q_R[]                             ''DQ1/Q_WX,RSRC/@1,ROT/ZERO,ALU/OR,MUX/R.S''
;4471   Q_R[]_M[]_0                       ''ALPCTL/WX_S.Q_R,SPW/MLONG,MSRC/@2,RSRC/@1,ROT/ZERO''
;4472   Q_R[]+0                           ''DQ1/Q_WX,RSRC/@1,ROT/ZERO,MUX/R.S,ALU/A+B+CI''
;4473   Q_R[]-Q                           ''DQ1/Q_WX,RSRC/@1,MUX/R.Q,ALU/A-B-CI''
;4474   Q_R[].SR.1                        ''DQ1/Q_WX,ALU/A+B+CI.SR,MUX/R.S,RSRC/@1,ROT/ZERO''
;4475   Q_R[]_D+RB+ALKC                   ''DQ1/Q_WX,RSRC/@1,SPW/RLONG,MUX/D.R1,ALU/A+B+CI,ALUCI/ALKC''
;4476   Q_SEXT(M[])                       ''DQ1/Q_WX,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALU/OR,ALUXM/SIGN''
;4477   Q_SEXT(XB) PC_PC+1                ''DQ1/Q_WX,MSRC/XB.PC_PC+1,RSRC/ZERO,MUX/XM.R,ALU/OR,ALUXM/SIGN,DTYPE/BYTE,
;4478                                          ISTRM/ISIZE_DSIZE,VSIZE/1''
;4479   Q_SEXT(XB)-R[] PC_PC+1            ''DQ1/Q_WX,MSRC/XB.PC_PC+1,RSRC/@1,ALU/A-B-CI,MUX/XM.R,ALUXM/SIGN,VSIZE/1,
;4480                                          DTYPE/BYTE,ISTRM/ISIZE_DSIZE''
;4481   Q_SEXT.M[] STEPC_[]               ''ALPCTL/WX_S.Q_XM,ALUXM/SIGN,ROT/ZLITO,LIT/LITRL,LITRL/@2,MSRC/@1,WCTRL/STEPC_WB''
;4482   Q_UNPACK(M[] R[])                 ''ALPCTL/WX_Q_S,MSRC/@1,RSRC/@2,ROT/GETFPF''
;4483   Q_XB PC_PC+1                      ''DQ1/Q_WX,MSRC/XB.PC_PC+1,RSRC/ZERO,MUX/M.R1,ALU/OR,ISTRM/ISIZE_DSIZE,DTYPE/IDEP,VSIZE/1''
;4484   Q_XB-R[] PC_PC+1                  ''DQ1/Q_WX,ALU/A-B-CI,MUX/M.R1,MSRC/XB.PC_PC+1,RSRC/@1,ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/BYTE''
;4485   Q_ZEXT(M[])                       ''DQ1/Q_WX,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALU/OR,ALUXM/ZERO''
;4486   Q_ZEXT(M[])-R[]                   ''DQ1/Q_WX,MSRC/@1,RSRC/@2,ALU/A-B-CI,MUX/XM.R,ALUXM/ZERO''
;4487   Q_ZEXT(M[]).SL.1                  ''DQ1/Q_WX,MSRC/@1,ROT/ZERO,MUX/XM.S,ALUXM/ZERO,ALU/A+B+CI.SL''
;4488   Q_ZEXT(XB)-R[] PC_PC+1            ''DQ1/Q_WX,ALU/A-B-CI,MUX/XM.R,MSRC/XB.PC_PC+1,RSRC/@1,ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/BYTE''
;4489   Q_ZLITO[]                         ''ALPCTL/WX_Q_S,ROT/ZLITO,LIT/LITRL,LITRL/@1''
;4490   Q_ZLITO[]-M[]                     ''DQ1/Q_WX,ROT/ZLITO,LIT/LITRL,LITRL/@1,MSRC/@2,MUX/M.S,ALU/B-A-CI''
;4491   Q_ZLIT4[]-M[]                     ''DQ1/Q_WX,ROT/ZLIT4,LIT/LITRL,LITRL/@1,MSRC/@2,MUX/M.S,ALU/B-A-CI''
;4492   Q_ZLITPL[]                        ''ALPCTL/WX_Q_S,ROT/ZLITPL,LIT/LITRL,LITRL/@1''
;4493
;4494   RBSP_0                            ''RSRC/ZERO.CLRRBSP''
;4495
;4496   RNUM_0                            ''MSRC/RNUM_WBUS,ROT/ZERO,ALPCTL/WX_S''
;4497   RNUM_1                            ''MSRC/RNUM_WBUS,ROT/MINUS1,MUX/Z.S,ALU/A-B-CI''
;4498   RNUM_D_D+1                        ''MSRC/RNUM_WBUS,DQ1/D_WX,ALU/A+B+CI,ALUCI/ONE,MUX/D.R1,RSRC/ZERO''
;4499   RNUM_D_D-1                        ''MSRC/RNUM_WBUS,DQ1/D_WX,ALU/A-B-CI,ALUCI/ONE,MUX/D.R1,RSRC/ZERO''
;4500   RNUM_D_ZLITO[]                    ''MSRC/RNUM_WBUS,DQ1/D_WX,ALU/A+B+CI,MUX/Z.S,ROT/ZLITO,LIT/LITRL,LITRL/@1''
;4501   RNUM_POPRBS_MTEMPO_POPSIZ         ''MSRC/READRBS,SPW/MLONG''
;4502   RNUM_Q_D_Q+1                      ''MSRC/RNUM_WBUS,DQ1/Q_D_WX,ALU/A+B+CI,MUX/R.Q,RSRC/ZERO,ALUCI/ONE''
;4503   RNUM_Q_Q+1                        ''MSRC/RNUM_WBUS,DQ1/Q_WX,ALU/A+B+CI,MUX/R.Q,RSRC/ZERO,ALUCI/ONE''
;4504   RNUM_Q_Q-1                        ''MSRC/RNUM_WBUS,DQ1/Q_WX,ALU/B-A-CI,MUX/R.Q,RSRC/ZERO,ALUCI/ONE''
;4505   RNUM_Q_ZLITO[]                    ''MSRC/RNUM_WBUS,ALPCTL/WX_Q_S,ROT/ZLITO,LIT/LITRL,LITRL/@1''
;4506   RNUM_R[]                          ''MSRC/RNUM_WBUS,RSRC/@1,ROT/ZERO,MUX/R.S,ALU/OR''
;4507   RNUM_R[]+CONX(2)                  ''MSRC/RNUM_WBUS,ALU/A+B+CI,MUX/R.S,RSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD''
;4508   RNUM_R[]_0                        ''MSRC/RNUM_WBUS,SPW/RLONG,RSRC/@1,ROT/ZERO,ALPCTL/WX_S''
;4509   RNUM_R[]_RB+1                     ''MSRC/RNUM_WBUS,SPW/RLONG,RSRC/@1,ALU/A-B-CI,MUX/R.S,ROT/MINUS1''
;4510   RNUM_R[]_RB+CONX(2)               ''MSRC/RNUM_WBUS,SPW/RLONG,RSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD,MUX/R.S,ALU/A+B+CI''
;4511   RNUM_R[]_RB-1                     ''MSRC/RNUM_WBUS,SPW/RLONG,RSRC/@1,ALU/A+B+CI,MUX/R.S,ROT/MINUS1''
;4512   RNUM_[]                           ''MSRC/RNUM_WBUS,ALPCTL/WX_S,ROT/ZLITO,LITRL/@1,LIT/LITRL''
;4513
;4514   RTEMPO_Q_-Q                       ''SPW/RLONG,RSRC/ZERO,DQ1/Q_WX,MUX/R.Q,ALU/A-B-CI''
```

```
;4515    RTEMP0_RNUM_PL_0                    'MSRC/RNUM_WBUS,ROT/SL.PL_WB,ALU/AND,MUX/R.S,RSRC/ZERO,SPW/RLONG''
;4516
;4517    RTEMP7_M[] PL<4-0>_31 PL<5>_1       'MSRC/@1,SPW/RLONG,ROT/OLIT0.PL_LIT,LIT/LITRL,LITRL/1FF,MUX/M.S,ALU/AND''
;4518    RTEMP7_M[]-ZLIT0[]                  'MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/A-B-CI,SPW/RLONG''
;4519    RTEMP7_M[].AND.ZLIT8[]              'MSRC/@1,ROT/ZLIT8,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/AND,SPW/RLONG''
;4520    RTEMP7_RNUM PL<4-0>_31 PL<5>_1      'ALPCTL/NOP,MSRC/WBUS_RNUM,SPW/RLONG,ROT/OLIT0.PL_LIT,LIT/LITRL,LITRL/1FF''
;4521    RTEMP7_RNUM_ZLIT0[]                 'SPW/RLONG,MSRC/RNUM_WBUS,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;4522
;4523    R[]                                 'RSRC/@1''
;4524    R[].SIZ_(RB+M[].ASR.3).SR.1         'RSRC/@1,SPW/RSIZE,ALU/A+B+CI.SR,MUX/R.S,ROT/ASR.M.3,MSRC/@2,DTYPE/BYTE,VSIZE/1''
;4525    R[].SIZ_-1                          'RSRC/@1,SPW/RSIZE,ROT/MINUS1,ALPCTL/WX_S''
;4526    R[].SIZ_-D                          'RSRC/@1,SPW/RSIZE,ROT/ZERO,MUX/D.S,ALU/B-A-CI''
;4527    R[].SIZ_-M[]                        'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/B-A-CI''
;4528    R[].SIZ_-RB                         'RSRC/@1,SPW/RSIZE,ROT/ZERO,MUX/R.S,ALU/B-A-CI''
;4529    R[].SIZ_0                           'RSRC/@1,SPW/RSIZE,ROT/ZERO,ALPCTL/WX_S''
;4530    R[].SIZ_D                           'RSRC/@1,SPW/RSIZE,ALU/OR,MUX/D.S,ROT/ZERO''
;4531    R[].SIZ_FLAGS                       'RSRC/@1,SPW/RSIZE,WCTRL/CM.TP.FPD.FLAGS''
;4532    R[].SIZ_FPA                         'RSRC/@1,SPW/RSIZE,FPA/WBUS_FPA''
;4533    R[].SIZ_M[]                         'RSRC/@1,SPW/RSIZE,ALU/OR,MUX/M.S,ROT/ZERO,MSRC/@2''
;4534    R[].SIZ_M[]+1                       'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.S,ROT/MINUS1,ALU/A-B-CI''
;4535    R[].SIZ_M[]+Q                       'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.Q1,ALU/A+B+CI''
;4536    R[].SIZ_M[]+Q+PSLC                  'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.Q1,ALU/A+B+CI,ALUCI/PSLC''
;4537    R[].SIZ_M[]+RB                      'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.R1,ALU/A+B+CI''
;4538    R[].SIZ_M[]+RB+PSLC                 'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.R1,ALU/A+B+CI,ALUCI/PSLC''
;4539    R[].SIZ_M[]-1                       'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.S,ROT/MINUS1,ALU/A+B+CI''
;4540    R[].SIZ_M[]-Q                       'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.Q1,ALU/A-B-CI''
;4541    R[].SIZ_M[].AND.RB                  'RSRC/@1,SPW/RSIZE,ALU/AND,MUX/M.R1,MSRC/@2''
;4542    R[].SIZ_M[].ANDNOT.Q                'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.Q1,ALU/ANDNOT''
;4543    R[].SIZ_M[].ASR.-P                  'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/ASR.M.-P,ALPCTL/WX_S''
;4544    R[].SIZ_M[].BCDSWP                  'RSRC/@1,SPW/RSIZE,MSRC/@2,ALPCTL/WX_S,ROT/BCDSWP''
;4545    R[].SIZ_M[].NOT                     'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/MINUS1,MUX/M.S,ALU/NOTAND''
;4546    R[].SIZ_M[].NOTAND.RB               'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.R1,ALU/NOTAND''
;4547    R[].SIZ_M[].OR.Q                    'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.Q1,ALU/OR''
;4548    R[].SIZ_M[].OR.RB                   'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.R1,ALU/OR''
;4549    R[].SIZ_M[].RR.8                    'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/RR.MM.SIZ,ALPCTL/WX_S,DTYPE/BYTE,VSIZE/1''
;4550    R[].SIZ_M[].RR.P                    'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/RR.MM.P,ALPCTL/WX_S''
;4551    R[].SIZ_M[].RR.SIZ                  'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/RR.MM.SIZ,ALPCTL/WX_S''
;4552    R[].SIZ_M[].SL.1                    'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/A+B+CI.SL''
;4553    R[].SIZ_M[].XOR.Q                   'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.Q1,ALU/XOR''
;4554    R[].SIZ_M[].XOR.RB                  'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.R1,ALU/XOR''
;4555    R[].SIZ_Q-RB-1                      'RSRC/@1,SPW/RSIZE,MUX/R.Q,ALU/B-A-CI,ALUCI/ONE''
;4556    R[].SIZ_Q Q_D                       'RSRC/@1,SPW/RSIZE,ALPCTL/WX_Q.Q_D''
;4557    R[].SIZ_Q_D_M[].ASL.P               'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/ASL.M.P,ALPCTL/WX_D_Q_S''
;4558    R[].SIZ_Q_M[]                       'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/OR,DQ1/Q_WX''
;4559    R[].SIZ_Q_M[].RR.SIZ                'RSRC/@1,SPW/RSIZE,MSRC/@2,ROT/RR.MM.SIZ,ALPCTL/WX_Q_S''
;4560    R[].SIZ_RB+1                        'RSRC/@1,SPW/RSIZE,ALU/A-B-CI,MUX/R.S,ROT/MINUS1''
;4561    R[].SIZ_RB+CONX(2)                  'RSRC/@1,SPW/RSIZE,ROT/CONX.SIZ,DTYPE/WORD,VSIZE/1,MUX/R.S,ALU/A+B+CI''
;4562    R[].SIZ_RB-CONX(2)                  'RSRC/@1,SPW/RSIZE,ROT/CONX.SIZ,DTYPE/WORD,VSIZE/1,MUX/R.S,ALU/A-B-CI''
;4563    R[].SIZ_RB+D                        'RSRC/@1,SPW/RSIZE,MUX/D.R1,ALU/A+B+CI''
;4564    R[].SIZ_RB-D                        'RSRC/@1,SPW/RSIZE,MUX/D.R1,ALU/B-A-CI''
;4565    R[].SIZ_RB-1                        'RSRC/@1,SPW/RSIZE,ROT/MINUS1,MUX/R.S,ALU/A+B+CI''
;4566    R[].SIZ_RB-M[]                      'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.R1,ALU/B-A-CI''
;4567    R[].SIZ_RB-M[]-PSLC                 'RSRC/@1,SPW/RSIZE,MSRC/@2,MUX/M.R1,ALU/B-A-CI,ALUCI/PSLC''
;4568    R[].SIZ_RB.SL.1                     'RSRC/@1,SPW/RSIZE,ROT/ZERO,MUX/R.S,ALU/A+B+CI.SL''
;4569
```

```
;4570   R[]_((M[] RB).RR.4.AND.MB).SL.1 ''RSRC/@1,SPW/RLONG,ALU/AND.SL,MUX/M.S,ROT/RR.MR.4,MSRC/@2''
;4571   R[]_(M[] RB).RR.4                ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RR.MR.4,ALPCTL/WX_S''
;4572   R[]_(M[] RB).RR.9                ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RR.MR.9,ALPCTL/WX_S''
;4573   R[]_(M[] RB).RR.P                ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RR.MR.P,ALPCTL/WX_S''
;4574   R[]_(M[] RB).RR.PS               ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RR.MR.PS,ALPCTL/WX_S''
;4575   R[]_(M[] RB).RR.S                ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RR.MR.S,ALPCTL/WX_S''
;4576   R[]_(M[] RB).XZ                  ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/XZ.MR,ALPCTL/WX_S''
;4577   R[]_(M[]+CONX(2)).SR.1           ''RSRC/@1,SPW/RLONG,ALU/A+B+CI.SR,MUX/M.S,MSRC/@2,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD''
;4578   R[]_(M[]+RB).BCD                 ''RSRC/@1,SPW/RLONG,ALU/A+B+CI.BCD,MUX/M.R1,VSIZE/1,DTYPE/LONG,MSRC/@2''
;4579   R[]_(M[]+RB+ALKC).BCD            ''RSRC/@1,SPW/RLONG,ALU/A+B+CI.BCD,ALUCI/ALKC,MUX/M.R1,VSIZE/1,DTYPE/LONG,MSRC/@2''
;4580   R[]_(M[]-RB).BCD                 ''RSRC/@1,SPW/RLONG,ALU/A-B-CI.BCD,MUX/M.R1,MSRC/@2,VSIZE/1,DTYPE/LONG''
;4581   R[]_(M[]-RB-ALKC).BCD            ''RSRC/@1,SPW/RLONG,ALU/A-B-CI.BCD,ALUCI/ALKC,MUX/M.R1,MSRC/@2,VSIZE/1,DTYPE/LONG''
;4582   R[]_(M[].AND.RB).RL.1            ''RSRC/@1,SPW/RLONG,ALU/AND.SL,MUX/M.R1,MSRC/@2,ALUSHF/ROT''
;4583   R[]_(RB M[]).RL.4                ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RL.RM.4,ALPCTL/WX_S''
;4584   R[]_(RB M[]).RL.P                ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RL.RM.P,ALPCTL/WX_S''
;4585   R[]_(RB+Q).SL.1                  ''RSRC/@1,SPW/RLONG,ALU/A+B+CI.SL,MUX/R.Q,ALUSHF/ZERO''
;4586   R[]_(RB-M[]<3-0>)                ''RSRC/@1,SPW/RLONG,ALU/A-B-CI,MUX/R.S,MSRC/@2,ROT/GETNIB''
;4587   R[]_(RB.OR.M[]<3-0>)             ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/R.S,MSRC/@2,ROT/GETNIB''
;4588   R[]_-1                           ''RSRC/@1,SPW/RLONG,ROT/MINUS1,ALPCTL/WX_S''
;4589   R[]_-D                           ''RSRC/@1,SPW/RLONG,ROT/ZERO,MUX/D.S,ALU/B-A-CI''
;4590   R[]_-M[]                         ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/B-A-CI''
;4591   R[]_-RB                          ''RSRC/@1,SPW/RLONG,ALU/B-A-CI,MUX/R.S,ROT/ZERO''
;4592   R[]_-ZEXT(M[])                   ''RSRC/@1,SPW/RLONG,MUX/XM.S,ROT/ZERO,ALU/B-A-CI,ALUXM/ZERO,MSRC/@2''
;4593   R[]_0                            ''RSRC/@1,SPW/RLONG,ROT/ZERO,ALPCTL/WX_S''
;4594   R[]_8                            ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG,ALU/A+B+CI.SL,MUX/Z.S''
;4595   R[]_CONX(1)                      ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE,ALPCTL/WX_S''
;4596   R[]_CONX(2)                      ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD,ALPCTL/WX_S''
;4597   R[]_D                            ''RSRC/@1,SPW/RLONG,ROT/ZERO,MUX/D.S,ALU/OR''
;4598   R[]_D+CONX(1)                    ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/D.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4599   R[]_D+CONX(2)                    ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/D.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD''
;4600   R[]_D+CONX(4)                    ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/D.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG''
;4601   R[]_D+RB                         ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/D.R1''
;4602   R[]_D+RB+ALKC                    ''RSRC/@1,SPW/RLONG,MUX/D.R1,ALU/A+B+CI,ALUCI/ALKC''
;4603   R[]_D-0                          ''RSRC/@1,SPW/RLONG,ROT/ZERO,MUX/D.S,ALU/A-B-CI''
;4604   R[]_D.NOTAND.RB                  ''RSRC/@1,SPW/RLONG,ALU/NOTAND,MUX/D.R1''
;4605   R[]_D.OR.(M[].RL.8)              ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/D.S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/LONG,MSRC/@2''
;4606   R[]_D.OR.RB                      ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/D.R1''
;4607   R[]_D_(D+RB).SL.1 Q<0>_1         ''RSRC/@1,SPW/RLONG,DQ3/SQL.D_WX,ALU/A+B+CI.SL,MUX/D.R2,ALUSHF/ALU0.Q1''
;4608   R[]_D_-D                         ''RSRC/@1,SPW/RLONG,ROT/ZERO,MUX/D.S,ALU/B-A-CI,DQ1/D_WX''
;4609   R[]_D_-M[]                       ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/B-A-CI,DQ1/D_WX''
;4610   R[]_D_0                          ''RSRC/@1,SPW/RLONG,ROT/ZERO,ALPCTL/WX_D_S''
;4611   R[]_D_D+CONX(4)                  ''RSRC/@1,SPW/RLONG,DQ1/D_WX,ALU/A+B+CI,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG,MUX/D.S''
;4612   R[]_D_D+Q+1                      ''RSRC/@1,SPW/RLONG,DQ1/D_WX,ALU/A+B+CI,MUX/D.Q1,ALUCI/ONE''
;4613   R[]_D_D-1                        ''RSRC/@1,SPW/RLONG,ROT/MINUS1,MUX/D.S,ALU/A+B+CI,DQ1/D_WX''
;4614   R[]_D_D-CONX(4)                  ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG,MUX/D.S,ALU/A-B-CI,DQ1/D_WX''
;4615   R[]_D_M[]                        ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/M.S,ROT/ZERO,MSRC/@2,DQ1/D_WX''
;4616   R[]_D_M[]-0                      ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/A-B-CI,DQ1/D_WX''
;4617   R[]_D_M[]-1                      ''RSRC/@1,SPW/RLONG,DQ1/D_WX,MSRC/@2,ROT/MINUS1,MUX/M.S,ALU/A+B+CI''
;4618   R[]_D_Q_D                        ''RSRC/@1,SPW/RLONG,ALPCTL/WX_D_Q.Q_D''
;4619   R[]_D_Q_-1                       ''RSRC/@1,SPW/RLONG,ALPCTL/WX_D_Q_S,ROT/MINUS1''
;4620   R[]_D_Q_RNUM_-1                  ''RSRC/@1,SPW/RLONG,ALPCTL/WX_D_Q_S,MSRC/RNUM_WBUS,ROT/MINUS1''
;4621   R[]_D_RB+1                       ''RSRC/@1,SPW/RLONG,ROT/MINUS1,MUX/R.S,ALU/A-B-CI,DQ1/D_WX''
;4622   R[]_D_RB+CONX(4)                 ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG,DQ1/D_WX''
;4623   R[]_D_RB-1                       ''RSRC/@1,SPW/RLONG,ROT/MINUS1,MUX/R.S,ALU/A+B+CI,DQ1/D_WX''
;4624   R[]_D_RB-CONX(4)                 ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG,MUX/R.S,ALU/A-B-CI,DQ1/D_WX''
```

```
;4625   R[]_D_ZEXT(M[])                  ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ZERO,MUX/XM.S,ALU/OR,DQ1/D_WX''
;4626   R[]_EXP(M[])                     ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/GETEXP,ALPCTL/WX_S''
;4627   R[]_FLAGS                        ''RSRC/@1,SPW/RLONG,WCTRL/CM.TP.FPD.FLAGS''
;4628   R[]_FLAGS_0                      ''RSRC/@1,SPW/RLONG,WCTRL/FLAGS_WB,ROT/ZERO,MUX/R.S,ALU/AND''
;4629   R[]_FPA                          ''RSRC/@1,SPW/RLONG,FPA/WBUS_FPA''
;4630   R[]_MEMSCR                       ''RSRC/@1,SPW/RLONG,WCTRL/MEMSCR''
;4631   R[]_M[]                          ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/M.S,ROT/ZERO,MSRC/@2''
;4632   R[]_M[]+(RB.ASL.SIZ)             ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ASL.R.SIZ,MUX/M.S,ALU/A+B+CI''
;4633   R[]_M[]+1                        ''RSRC/@1,SPW/RLONG,ALU/A-B-CI,MUX/M.S,MSRC/@2,ROT/MINUS1''
;4634   R[]_M[]+CONX(1)                  ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/CONX.SIZ.VSIZE/1,DTYPE/BYTE,MUX/M.S,ALU/A+B+CI''
;4635   R[]_M[]+CONX(2)                  ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD,MUX/M.S,ALU/A+B+CI''
;4636   R[]_M[]+CONX(4)                  ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG,MUX/M.S,ALU/A+B+CI''
;4637   R[]_M[]+PL                       ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/PL,MUX/M.S,ALU/A+B+CI''
;4638   R[]_M[]+Q                        ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.Q1,ALU/A+B+CI''
;4639   R[]_M[]+Q+1                      ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,ALUCI/ONE,MUX/M.Q1,MSRC/@2''
;4640   R[]_M[]+RB                       ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/M.R1,MSRC/@2''
;4641   R[]_M[]+RB+ALKC                  ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.R1,ALU/A+B+CI,ALUCI/ALKC''
;4642   R[]_M[]+RB+PSLC                  ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.R1,ALU/A+B+CI,ALUCI/PSLC''
;4643   R[]_M[]+SL                       ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/SL,MUX/M.S,ALU/A+B+CI''
;4644   R[]_M[]-0                        ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/A-B-CI''
;4645   R[]_M[]-1                        ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/M.S,MSRC/@2,ROT/MINUS1''
;4646   R[]_M[]-CONX(2)                  ''RSRC/@1,SPW/RLONG,MSRC/@2,VSIZE/1,DTYPE/WORD,ROT/CONX.SIZ,MUX/M.S,ALU/A-B-CI''
;4647   R[]_M[]-CONX(4)                  ''RSRC/@1,SPW/RLONG,MSRC/@2,VSIZE/1,DTYPE/LONG,ROT/CONX.SIZ,MUX/M.S,ALU/A-B-CI''
;4648   R[]_M[]-Q                        ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.Q1,ALU/A-B-CI''
;4649   R[]_M[]-RB                       ''RSRC/@1,SPW/RLONG,ALU/A-B-CI,MUX/M.R1,MSRC/@2''
;4650   R[]_M[].AND.RB                   ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.R1,ALU/AND''
;4651   R[]_M[].ANDNOT.Q                 ''RSRC/@1,SPW/RLONG,MSRC/@2,ALU/ANDNOT,MUX/M.Q1''
;4652   R[]_M[].ASL.P                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ASL.M.P,ALPCTL/WX_S''
;4653   R[]_M[].ASR.-P                   ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ASR.M.-P,ALPCTL/WX_S''
;4654   R[]_M[].ASR.3                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ASR.M.3,ALPCTL/WX_S''
;4655   R[]_M[].ASR.P                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ASR.M.P,ALPCTL/WX_S''
;4656   R[]_M[].BCDSWP                   ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/BCDSWP,ALPCTL/WX_S''
;4657   R[]_M[].CLR1B                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/CLR1BM,ALPCTL/WX_S''
;4658   R[]_M[].CLR2B                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/CLR2BM,ALPCTL/WX_S''
;4659   R[]_M[].CLR3B                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/CLR3BM,ALPCTL/WX_S''
;4660   R[]_M[].FPLIT                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/FPLIT,ALPCTL/WX_S''
;4661   R[]_M[].NIBBLE                   ''RSRC/@1,SPW/RLONG,ROT/GETNIB,MSRC/@2,ALPCTL/WX_S''
;4662   R[]_M[].OR.(RB.RR.16)            ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/M.S,ROT/RR.RR.SIZ,MSRC/@2,VSIZE/1,DTYPE/WORD''
;4663   R[]_M[].OR.(RB.RL.16)            ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/M.S,ROT/RR.RR.SIZ,MSRC/@2,VSIZE/1,DTYPE/WORD''
;4664   R[]_M[].OR.(RB.RR.SIZ)           ''RSRC/@1,SPW/RLONG,ROT/RR.RR.SIZ,MSRC/@2,ALU/OR,MUX/M.S''
;4665   R[]_M[].OR.Q                     ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/M.Q1,MSRC/@2''
;4666   R[]_M[].OR.RB                    ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.R1,ALU/OR''
;4667   R[]_M[].RL.24                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE,ALPCTL/WX_S''
;4668   R[]_M[].RL.9                     ''RSRC/@1,SPW/RLONG,MSRC/@2,ALPCTL/WX_S,ROT/RL.MM.PTE''
;4669   R[]_M[].RL.P                     ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RL.MM.P,ALPCTL/WX_S''
;4670   R[]_M[].RR.16                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,ALPCTL/WX_S''
;4671   R[]_M[].RR.24                    ''RSRC/@1,SPW/RLONG,ALPCTL/WX_S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/LONG,MSRC/@2''
;4672   R[]_M[].RR.8                     ''RSRC/@1,SPW/RLONG,ALPCTL/WX_S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE,MSRC/@2''
;4673   R[]_M[].RR.P                     ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RR.MM.P,ALPCTL/WX_S''
;4674   R[]_M[].RR.PS                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/RR.MM.PS,ALPCTL/WX_S''
;4675   R[]_M[].SR.1                     ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.S,ROT/ZERO,ALU/A+B+CI.SR''
;4676   R[]_M[].XOR.RB                   ''RSRC/@1,SPW/RLONG,ALU/XOR,MUX/M.R1,MSRC/@2''
;4677   R[]_M[].XZ                       ''RSRC/@1,SPW/RLONG,ALPCTL/WX_S,ROT/XZ.MM,MSRC/@2''
;4678   R[]_M[]<3-0>+RB                  ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/R.S,MSRC/@2,ROT/GETNIB''
;4679   R[]_NOT(M[]).AND.RB              ''RSRC/@1,SPW/RLONG,ALU/NOTAND,MUX/M.R1,MSRC/@2''
```

```
;4680   R[]_NOT(M[].ASR.-P)            ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ASR.M.-P,ALPCTL/WX_.NOT.S''
;4681   R[]_PL_M[]-RB                  ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.R1,ALU/A-B-CI,ROT/SL.PL_WB''
;4682   R[]_PL_Q Q_M[]                 ''RSRC/@1,SPW/RLONG,ROT/SL.PL_WB,MSRC/@2,ALPCTL/WX_Q.Q_M''
;4683   R[]_PL_Q-M[]                   ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.Q1,ALU/B-A-CI,ROT/SL.PL_WB''
;4684   R[]_PS[                        ''RSRC/@1,SPW/RLONG,CCPSL/WB_PSL.CCBR_SIGND''
;4685   R[]_Q Q_D                      ''RSRC/@1,SPW/RLONG,ALPCTL/WX_Q.Q_D''
;4686   R[]_Q Q_M[]                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ALPCTL/WX_Q.Q_M''
;4687   R[]_Q-M[]                      ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.Q1,ALU/B-A-CI''
;4688   R[]_Q_((M[]+Q).SL.1).OR.1      ''RSRC/@1,SPW/RLONG,DQ1/Q_WX,ALU/A+B+CI.SL,MUX/M.Q1,MSRC/@2,ALUSHF/ONE''
;4689   R[]_Q_0                        ''RSRC/@1,SPW/RLONG,ROT/ZERO,ALPCTL/WX_Q_S''
;4690   R[]_Q_CONX(1)                  ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE,ALPCTL/WX_Q_S''
;4691   R[]_Q_CONX(4)                  ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG,ALPCTL/WX_Q_S''
;4692   R[]_Q_D                        ''RSRC/@1,SPW/RLONG,ROT/ZERO,MUX/D.S,ALU/OR,DQ1/Q_WX''
;4693   R[]_Q_D+RB+ALKC                ''RSRC/@1,SPW/RLONG,DQ1/Q_WX,MUX/D.R1,ALU/A+B+CI,ALUCI/ALKC''
;4694   R[]_Q_D_M[]                    ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/OR,DQ1/Q_D_WX''
;4695   R[]_Q_EXP(M[])                 ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/GETEXP,ALU/A+B+CI,MUX/Z.S,DQ1/Q_WX''
;4696   R[]_Q_M[]                      ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/OR,DQ1/Q_WX''
;4697   R[]_Q_M[].XZ.MM                ''RSRC/@1,SPW/RLONG,ALPCTL/WX_Q_S,MSRC/@2,ROT/XZ.MM''
;4698   R[]_Q_Q-M[]                    ''RSRC/@1,SPW/RLONG,DQ1/Q_WX,ALU/B-A-CI,MUX/M.Q1,MSRC/@2''
;4699   R[]_Q_RB+Q                     ''RSRC/@1,SPW/RLONG,DQ1/Q_WX,MUX/R.Q,ALU/A+B+CI''
;4700   R[]_Q_ZEXT(M[])                ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ZERO,MUX/XM.S,ALU/OR DQ1/Q_WX''
;4701   R[]_RB+1                       ''RSRC/@1,SPW/RLONG,ROT/MINUS1,MUX/R.S,ALU/A-B-CI''
;4702   R[]_RB+CONX(1)                 ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4703   R[]_RB+CONX(2)                 ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,MUX/R.S,ALU/A+B+CI,VSIZE/1,DTYPE/WORD''
;4704   R[]_RB+CONX(4)                 ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG''
;4705   R[]_RB+CONX.SIZ                ''RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/IDEP''
;4706   R[]_RB+Q                       ''RSRC/@1,SPW/RLONG,MUX/R.Q,ALU/A+B+CI''
;4707   R[]_RB-1                       ''RSRC/@1,SPW/RLONG,ROT/MINUS1,MUX/R.S,ALU/A+B+CI''
;4708   R[]_RB-CONX(2)                 ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,MUX/R.S,ALU/A-B-CI,VSIZE/1,DTYPE/WORD''
;4709   R[]_RB-CONX(4)                 ''RSRC/@1,SPW/RLONG,ALU/A-B-CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG''
;4710   R[]_RB-CONX.SIZ                ''RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,MUX/R.S,ALU/A-B-CI''
;4711   R[]_RB-D-ALKC                  ''RSRC/@1,SPW/RLONG,MUX/D.R1,ALU/B-A-CI,ALUCI/ALKC''
;4712   R[]_RB-M[]                     ''RSRC/@1,SPW/RLONG,ALU/B-A-CI,MUX/M.R1,MSRC/@2''
;4713   R[]_RB-M[]-PSLC                ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.R1,ALU/B-A-CI,ALUCI/PSLC''
;4714   R[]_RB-Q                       ''RSRC/@1,SPW/RLONG,MUX/R.Q,ALU/A-B-CI''
;4715   R[]_RB.AND.(M[].RR.16)         ''RSRC/@1,SPW/RLONG,ALU/AND,MUX/R.S,ROT/RR.MM.SIZ,MSRC/@2,VSIZE/1,DTYPE/WORD''
;4716   R[]_RB.AND.(M[].RR.8)          ''RSRC/@1,SPW/RLONG,ALU/AND,MUX/R.S,ROT/RR.MM.SIZ,MSRC/@2,VSIZE/1,DTYPE/BYTE''
;4717   R[]_RB.ANDNOT.Q                ''RSRC/@1,SPW/RLONG,MUX/R.Q,ALU/ANDNOT''
;4718   R[]_RB.ASL.1                   ''RSRC/@1,SPW/RLONG,ALPCTL/WX_S,ROT/ASL.R.SIZ,VSIZE/1,DTYPE/WORD''
;4719   R[]_RB.ASL.2                   ''RSRC/@1,SPW/RLONG,ALPCTL/WX_S,ROT/ASL.R.SIZ,VSIZE/1,DTYPE/LONG''
;4720   R[]_RB.ASL.7                   ''RSRC/@1,SPW/RLONG,ALPCTL/WX_S,ROT/ASL.R.7''
;4721   R[]_RB.ASL.SIZ                 ''RSRC/@1,SPW/RLONG,ALPCTL/WX_S,ROT/ASL.R.SIZ''
;4722   R[]_RB.OR.(M[].ASL.P)          ''RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ASL.M.P,MUX/R.S,ALU/OR''
;4723   R[]_RB.OR.(M[].RL.16)          ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/R.S,ROT/RR.MM.SIZ,MSRC/@2,VSIZE/1,DTYPE/WORD''
;4724   R[]_RB.OR.(M[].RL.24)          ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/R.S,ROT/RR.MM.SIZ,MSRC/@2,VSIZE/1,DTYPE/BYTE''
;4725   R[]_RB.OR.(M[].RL.8)           ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/R.S,ROT/RR.MM.SIZ,MSRC/@2,VSIZE/1,DTYPE/LONG''
;4726   R[]_RB.OR.(M[].RL.9)           ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/R.S,ALU/OR,ROT/RL.MM.PTE''
;4727   R[]_RB.OR.(M[].RR.16)          ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/R.S,MSRC/@2,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD''
;4728   R[]_RB.OR.(M[].RR.24)          ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/R.S,MSRC/@2,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/LONG''
;4729   R[]_RB.OR.(M[].RR.8)           ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/R.S,MSRC/@2,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE''
;4730   R[]_RB.OR.(M[].RR.SIZ)         ''RSRC/@1,SPW/RLONG,ROT/RR.MM.SIZ,MUX/R.S,ALU/OR,MSRC/@2''
;4731   R[]_RB.OR.(M[].XZ)             ''RSRC/@1,SPW/RLONG,ROT/XZ.MM,MUX/R.S,ALU/OR,MSRC/@2''
;4732   R[]_RB.OR.CONX(1)              ''RSRC/@1,SPW/RLONG,ALU/OR,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4733   R[]_RB.OR.Q                    ''RSRC/@1,SPW/RLONG,MUX/R.Q,ALU/OR''
;4734   R[]_RB.OR.ZEXT(M[])            ''RSRC/@1,SPW/RLONG,MSRC/@2,MUX/XM.R,ALUXM/ZERO,ALU/OR''
```

```
;4735   R[]_RB.RL.16                    "RSRC/@1,SPW/RLONG,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/WORD,ALPCTL/WX_S"
;4736   R[]_RB.RL.8                     "RSRC/@1,SPW/RLONG,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/LONG,ALPCTL/WX_S"
;4737   R[]_RB.RL.P                     "RSRC/@1,SPW/RLONG,ROT/RL.RR.P,ALPCTL/WX_S"
;4738   R[]_RB.RR.24                    "RSRC/@1,SPW/RLONG,ALPCTL/WX_S,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/LONG"
;4739   R[]_RB.RR.8                     "RSRC/@1,SPW/RLONG,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/BYTE,ALPCTL/WX_S"
;4740   R[]_RB.RR.P                     "RSRC/@1,SPW/RLONG,ROT/RR.RR.P,ALPCTL/WX_S"
;4741   R[]_RB.XOR.MINUS1               "RSRC/@1,SPW/RLONG,ROT/MINUS1,MUX/R.S,ALU/XOR"
;4742   R[]_RB.XZ                       "RSRC/@1,SPW/RLONG,ALPCTL/WX_S,ROT/XZ.RR"
;4743   R[]_RBSP                        "RSRC/@1,SPW/RLONG,MSRC/WB_RBSP,ALPCTL/NOP"
;4744   R[]_RNUM                        "RSRC/@1,SPW/RLONG,ALPCTL/NOP,MSRC/WBUS_RNUM"
;4745   R[]_RNUM_MTEMP0.XZ              "RSRC/@1,SPW/RLONG,MSRC/RNUM_WBUS,ALPCTL/WX_S,ROT/XZ.MM"
;4746   R[]_SEXT(M[])                   "RSRC/@1,SPW/RLONG,MSRC/@2,ALUXM/SIGN,MUX/XM.S,ROT/MINUS1,ALU/AND"
;4747   R[]_TB                          "RSRC/@1,SPW/RLONG,MSRC/TB,BUS/PRB.RD,VSIZE/1,ALU/OR,MUX/M.S,ROT/ZERO"
;4748   R[]_VA_M[]                      "RSRC/@1,SPW/RLONG,MSRC/@2,ROT/ZERO,MUX/M.S,ALU/OR,WCTRL/VA_WB"
;4749   R[]_VA_RB+1                     "RSRC/@1,SPW/RLONG,ROT/MINUS1,MUX/R.S,ALU/A-B-CI,WCTRL/VA_WB"
;4750   R[]_VA_RB-Q                     "RSRC/@1,SPW/RLONG,MUX/R.Q,ALU/A-B-CI,WCTRL/VA_WB"
;4751   R[]_WDR                         "RSRC/@1,SPW/RLONG,WCTRL/MBUS_WDR,MSRC/WDR,ALU/OR,MUX/M.S,ROT/ZERO"
;4752   R[]_XB PC_PC+1                  "RSRC/@1,SPW/RLONG,ALU/OR,MUX/M.S,MSRC/XB.PC_PC+I,ROT/ZERO,ISTRM/ISIZE_DSIZE,
;4753                                                   VSIZE/1,DTYPE/BYTE"
;4754   R[]_XB PC_PC+4                  "RSRC/@1,SPW/RLONG,ALU/OR,MUX/M.S,MSRC/XB.PC_PC+I,ROT/ZERO,ISTRM/ISIZE_DSIZE,
;4755                                                   VSIZE/1,DTYPE/LONG"
;4756   R[]_XB PC_PC+I                  "RSRC/@1,SPW/RLONG,ALU/OR,MUX/M.S,MSRC/XB.PC_PC+I,ROT/ZERO,ISTRM/ISIZE_DSIZE,
;4757                                                   VSIZE/1,DTYPE/IDEP"
;4758   R[]_ZEXT(M[])                   "RSRC/@1,SPW/RLONG,ALU/OR,MUX/XM.S,MSRC/@2,ROT/ZERO"
;4759   R[]_ZEXT(M[]).SL.1              "RSRC/@1,SPW/RLONG,ALU/A+B+CI.SL,MUX/XM.S,ALUXM/ZERO,MSRC/@2,ROT/ZERO"
;4760   R[]_ZEXT(XB) PC_PC+1            "RSRC/@1,SPW/RLONG,MSRC/XB.PC_PC+I,ROT/ZERO,MUX/XM.S,ALU/OR,ISTRM/ISIZE_DSIZE,
;4761                                                   VSIZE/1,DTYPE/BYTE"
;4762   R[]_ZEXT(XB) PC_PC+2            "RSRC/@1,SPW/RLONG,MSRC/XB.PC_PC+I,ROT/ZERO,MUX/XM.S,ALU/OR,ISTRM/ISIZE_DSIZE,
;4763                                                   VSIZE/1,DTYPE/WORD"
;4764
;4765   SL_[]                           "ROT/OLIT0.SL_LIT,LIT/LITRL,LITRL/@1"
;4766   SOFTIPR_0                       "WCTRL/SOFTIPR_WB,ROT/ZERO,ALPCTL/WX_S"
;4767   SOFTIPR_M[].RR.16               "WCTRL/SOFTIPR_WB,ALPCTL/WX_S,MSRC/@1,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD"
;4768   SPICR_SPNICR                    "RSRC/SPNICR.SPICR,SPW/RSIZE,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/WORD,ALPCTL/WX_S"
;4769
;4770   STEPC_(M[]+2).SR.1              "WCTRL/STEPC_WB,ALU/A+B+CI.SR,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/2"
;4771   STEPC_(M[]+ZLIT0[]).SR.1        "WCTRL/STEPC_WB,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,ALU/A+B+CI.SR,MUX/M.S"
;4772   STEPC_(SEXT(M[])+2).SR.1        "WCTRL/STEPC_WB,ALU/A+B+CI.SR,MUX/XM.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/2"
;4773   STEPC_(ZEXT(M[])+CONX(2)).SR.1  "WCTRL/STEPC_WB,ALU/A+B+CI.SR,MUX/XM.S,MSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD,ALUXM/ZERO"
;4774   STEPC_14.                       "MISC/SC_14"
;4775   STEPC_2                         "MISC/SC_2"
;4776   STEPC_30.                       "MISC/SC_30"
;4777   STEPC_6                         "MISC/SC_6"
;4778   STEPC_D_(M[]+CONX(2)).SR.1      "WCTRL/STEPC_WB,DQ1/D_WX,ALU/A+B+CI.SR,MUX/M.S,MSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD"
;4779   STEPC_D_M[]-ZLIT0[]             "WCTRL/STEPC_WB,DQ1/D_WX,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/A-B-CI"
;4780   STEPC_D_M[].SR.1                "WCTRL/STEPC_WB,DQ1/D_WX,ALU/A+B+CI.SR,MUX/M.R1,MSRC/@1,RSRC/ZERO"
;4781   STEPC_D_RNUM_ZLIT0[]            "WCTRL/STEPC_WB,ALPCTL/WX_D_S,MSRC/RNUM_WBUS,ROT/ZLIT0,LIT/LITRL,LITRL/@1"
;4782   STEPC_D_SEXT(M[]).SR.1          "WCTRL/STEPC_WB,DQ1/D_WX,ALU/A+B+CI.SR,MUX/XM.R,ALUXM/SIGN,MSRC/@1,RSRC/ZERO"
;4783   STEPC_M[]+ZLIT0[]               "WCTRL/STEPC_WB,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,ALU/A+B+CI,MUX/M.S"
;4784   STEPC_M[]-R[]                   "WCTRL/STEPC_WB,MSRC/@1,RSRC/@2,ALU/A-B-CI,MUX/M.R1"
;4785   STEPC_M[]-ZLIT0[]               "WCTRL/STEPC_WB,MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT0,ALU/A-B-CI,MUX/M.S"
;4786   STEPC_M[].SR.1                  "WCTRL/STEPC_WB,ALU/A+B+CI.SR,MUX/M.S,MSRC/@1,ROT/ZERO"
;4787   STEPC_M[]<3-0>.                 "WCTRL/STEPC_WB,MSRC/@1,ROT/GETNIB,ALPCTL/WX_S"
;4788   STEPC_M[]_R[].RR.P              "WCTRL/STEPC_WB,MSRC/@1,SPW/MLONG,ALPCTL/WX_S,RSRC/@2,ROT/RR.RR.P"
;4789   STEPC_M[]_ZLIT0[]               "WCTRL/STEPC_WB,MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@2"
```

```
;4790   STEPC_M[]_ZLIT28[]              'WCTRL/STEPC_WB,MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT28,ALPCTL/WX_S''
;4791   STEPC_PL                        'WCTRL/STEPC_WB,ROT/PL,ALPCTL/WX_S''
;4792   STEPC_Q_M[].SR.1                'WCTRL/STEPC_WB,DQ1/Q_WX,ALU/AND.SR,MUX/M.S,MSRC/@1,ROT/MINUS1''
;4793   STEPC_Q_R[].SR.1                'WCTRL/STEPC_WB,DQ1/Q_WX,ALU/AND.SR,MUX/R.S,RSRC/@1,ROT/MINUS1''
;4794   STEPC_R[].RR.P                  'WCTRL/STEPC_WB,RSRC/@1,ROT/RR.RR.P,ALPCTL/WX_S''
;4795   STEPC_R[].SR.1                  'WCTRL/STEPC_WB,ALU/A+B+CI.SR,MUX/R.S,RSRC/@1,ROT/ZERO''
;4796   STEPC_ZLIT0[]                   'WCTRL/STEPC_WB,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;4797
;4798   TB_BAD PARITY                   'WCTRL/TB_WB,MISC/FORCE.TB,MSRC/VA,ALPCTL/WX_S,ROT/MINUS1''
;4799   TB_D+ZLIT8[]                    'WCTRL/TB_WB,ALU/A+B+CI,MUX/D.S,ROT/ZLIT8,LIT/LITRL,LITRL/@1,MSRC/VA''
;4800   TB_R[]                          'WCTRL/TB_WB,ALU/OR,MUX/R.S,RSRC/@1,ROT/ZERO,MSRC/VA''
;4801
;4802   TCSR_0                          'WCTRL/TCSR_WB,ROT/ZERO,ALPCTL/WX_S''
;4803   TCSR_M[].AND.ZLIT16[]           'WCTRL/TCSR_WB,ALU/AND,MUX/M.S,MSRC/@1,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4804   TCSR_M[].OR.R[]                 'WCTRL/TCSR_WB,ALU/OR,MUX/M.R1,MSRC/@1,RSRC/@2''
;4805
;4806   TODCR_D.XOR.ZLIT16[]            'WCTRL/TODCLK_WB,ALU/XOR,MUX/D.S,ROT/ZLIT16,LIT/LITRL,LITRL/@1''
;4807   TODCR_D_(M[].XZ).OR.R[]         'WCTRL/TODCLK_WB,DQ1/D_WX,ALU/OR,MUX/R.S,RSRC/@2,ROT/XZ.MM,MSRC/@1''
;4808   TODCR_M[].OR.R[]                'WCTRL/TODCLK_WB,ALU/OR,MUX/M.R1,MSRC/@1,RSRC/@2''
;4809   TODCR_M[].OR.ZLIT16[]           'WCTRL/TODCLK_WB,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4810   TODCR_M[]_MB-ZLIT16[]           'WCTRL/TODCLK_WB,MSRC/@1.SPW/MLONG,ALU/A-B-CI,MUX/M.S,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4811   TODCR_M[]_ZLIT16[]              'WCTRL/TODCLK_WB,SPW/MLONG,MSRC/@1,ALPCTL/WX_S,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;4812   TRAR_ZLIT16[]                   'WCTRL/LOADTRAR,ALPCTL/WX_S,ROT/ZLIT16,LIT/LITRL,LITRL/@1''
;4813
;4814   TU58REGS_M[].AND.OLIT16[]       'WCTRL/TU58WRITE,MSRC/@1,ROT/OLIT16,LIT/LITRL,LITRL/@2,ALU/AND,MUX/M.S''
;4815   TU58REGS_M[].OR.ZLIT16[]        'WCTRL/TU58WRITE,MSRC/@1,ROT/ZLIT16,LIT/LITRL,LITRL/@2,ALU/OR,MUX/M.S''
;4816   TU58REGS_M[].RR.16              'WCTRL/TU58WRITE,ALPCTL/WX_S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,MSRC/@1''
;4817   TU58REGS_M[]_MB.RL.8            'WCTRL/TU58WRITE,MSRC/@1,SPW/MLONG,ALPCTL/WX_S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/LONG''
;4818   TU58REGS_R[]_0                  'WCTRL/TU58WRITE,SPW/RLONG,RSRC/@1,ALPCTL/WX_S,ROT/ZERO''
;4819   TU58REGS_ZLIT16[]               'WCTRL/TU58WRITE,ALPCTL/WX_S,ROT/ZLIT16,LIT/LITRL,LITRL/@1''
;4820
;4821   VA_D+R[]                        'WCTRL/VA_WB,ALU/A+B+CI,MUX/D.R1,RSRC/@1''
;4822   VA_D+R[]+1                      'WCTRL/VA_WB,ALU/A+B+CI,MUX/D.R1,RSRC/@1,ALUCI/ONE''
;4823   VA_D+ZLIT0[]                    'WCTRL/VA_WB,ALU/A+B+CI,MUX/D.S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;4824   VA_D-ZLIT0[]                    'WCTRL/VA_WB,LIT/LITRL,LITRL/@1,ROT/ZLIT0,MUX/D.S,ALU/A-B-CI''
;4825   VA_D.XOR.ZLIT16[]               'WCTRL/VA_WB,ALU/XOR,MUX/D.S,ROT/ZLIT16,LIT/LITRL,LITRL/@1''
;4826   VA_D_D+ZLIT8[] INVALIDATE TB    'WCTRL/CLRTB.VA_WB,DQ1/D_WX,ALU/A+B+CI,MUX/D.S,ROT/ZLIT8,LIT/LITRL,LITRL/@1,BUS/PRB.RD.PTE''
;4827   VA_D_M[]+R[]                    'WCTRL/VA_WB,DQ1/D_WX,ALU/A+B+CI,MUX/M.R1,MSRC/@1,RSRC/@2''
;4828   VA_D_M[]-ZLIT0[]                'WCTRL/VA_WB,DQ1/D_WX,ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4829   VA_D_M[]_R[]                    'WCTRL/VA_WB,DQ1/D_WX,MSRC/@1,SPW/MLONG,RSRC/@2,MUX/R.S,ROT/ZERO,ALU/OR''
;4830   VA_D_ZLIT0[]                    'WCTRL/VA_WB,ALPCTL/WX_D_S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;4831   VA_D_ZLIT24[]                   'WCTRL/VA_WB,ALPCTL/WX_D_S,ROT/ZLIT24,LIT/LITRL,LITRL/@1''
;4832   VA_MTEMP0_D_VA+ZLIT8[]          'WCTRL/VA_WB,SPW/MLONG,DQ1/D_WX,MSRC/VA,ROT/ZLIT8,LIT/LITRL,LITRL/@1,MUX/M.S,ALU/A+B+CI''
;4833   VA_MTEMP0_PC                    'WCTRL/VA_WB,MSRC/PC,SPW/MLONG,RSRC/ZERO,MUX/M.R1,ALU/OR''
;4834   VA_MTEMP0_PC+SEXT(XB)+2 PC_PC+2 'WCTRL/VA_PC+I+W.PC_PC+I,MSRC/XB.PC_PC+I,SPW/MLONG,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/OR,
;4835                                                    VSIZE/1,DTYPE/WORD,ISTRM/ISIZE_DSIZE''
;4836   VA_MTEMP0_VA.AND.OLIT0[]        'WCTRL/VA_WB,MSRC/VA,ROT/OLIT0,LIT/LITRL,LITRL/@1,MUX/M.S,ALU/AND,SPW/MLONG''
;4837   VA_MTEMP0_SEXT(XB)+R[] PC_PC+2  'WCTRL/VA_WB,MSRC/XB.PC_PC+I,SPW/MLONG,RSRC/@1,MUX/XM.R,ALUXM/SIGN,ALU/A+B+CI,
;4838                                                    VSIZE/1,DTYPE/WORD,ISTRM/ISIZE_DSIZE''
;4839   VA_MTEMP0_ZEXT(MDR)             'WCTRL/VA_WB,MSRC/MDR,SPW/MLONG,RSRC/ZERO,MUX/XM.R,ALUXM/ZERO,ALU/OR''
;4840   VA_MTEMP0_ZEXT(XB) PC_PC+2      'WCTRL/VA_WB,MSRC/XB.PC_PC+I,SPW/MLONG,RSRC/ZERO,MUX/XM.R,ALUXM/ZERO,ALU/OR,
;4841                                                    ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/WORD''
;4842   VA_M[]                          'WCTRL/VA_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/OR''
;4843   VA_M[]+(R[].ASL.SIZE)           'WCTRL/VA_WB,MSRC/@1,RSRC/@2,ROT/ASL.R.SIZ,MUX/M.S,ALU/A+B+CI''
;4844   VA_M[]-(R[].RR.8)               'WCTRL/VA_WB,ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/RR.RR.SIZ,RSRC/@2,VSIZE/1,DTYPE/BYTE''
```

```
;4845   VA_M[]+CONX(1)                          'WCTRL/VA_WB,MSRC/@1,VSIZE/1,DTYPE/BYTE,ROT/CONX.SIZ,MUX/M.S,ALU/A+B+CI''
;4846   VA_M[]+CONX(2)                          'WCTRL/VA_WB,MSRC/@1,VSIZE/1,DTYPE/WORD,ROT/CONX.SIZ,MUX/M.S,ALU/A+B+CI''
;4847   VA_M[]+CONX(4)                          'WCTRL/VA_WB,MSRC/@1,VSIZE/1,DTYPE/LONG,ROT/CONX.SIZ,MUX/M.S,ALU/A+B+CI''
;4848   VA_M[]+Q+1                              'WCTRL/VA_WB,ALU/A+B+CI,ALUCI/ONE,MUX/M.Q1,MSRC/@1''
;4849   VA_M[]+R[]                              'WCTRL/VA_WB,ALU/A+B+CI,MUX/M.R1,MSRC/@1,RSRC/@2''
;4850   VA_M[]+R[]+1                            'WCTRL/VA_WB,ALU/A+B+CI,MUX/M.R1,MSRC/@1,RSRC/@2,ALUCI/ONE''
;4851   VA_M[]+ZLIT0[]                          'WCTRL/VA_WB,ALU/A+B+CI,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4852   VA_M[]+ZLIT8[]                          'WCTRL/VA_WB,MSRC/@1,ROT/ZLIT8,LIT/LITRL,LITRL/@2,MUX/M.S,ALU/A+B+CI''
;4853   VA_M[]-(R[].ASL.SIZE)                   'WCTRL/VA_WB,MSRC/@1,RSRC/@2,ROT/ASL.R.SIZ,MUX/M.S,ALU/A-B-CI''
;4854   VA_M[]-R[]                              'WCTRL/VA_WB,ALU/A-B-CI,MUX/M.R1,MSRC/@1,RSRC/@2''
;4855   VA_M[]-ZLIT0[]                          'WCTRL/VA_WB,ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4856   VA_M[].AND.ZLIT0[]                      'WCTRL/VA_WB,ALU/AND,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4857   VA_M[].ANDNOT.ZLIT0[]                   'WCTRL/VA_WB,MSRC/@1,ROT/ZLIT0,MUX/M.S,ALU/ANDNOT,LITRL/@2,LIT/LITRL''
;4858   VA_M[].OR.Q                             'WCTRL/VA_WB,ALU/OR,MUX/M.Q1,MSRC/@1''
;4859   VA_M[].XOR.R[]                          'WCTRL/VA_WB,ALU/XOR,MUX/M.R1,MSRC/@1,RSRC/@2''
;4860   VA_M[].XOR.ZLIT0[]                      'WCTRL/VA_WB,ALU/XOR,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4861   VA_M[].XOR.ZLIT24[]                     'WCTRL/VA_WB,ALU/XOR,MUX/M.S,MSRC/@1,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;4862   VA_M[]_MB+ZLIT0[]                       'WCTRL/VA_WB,MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4863   VA_M[]_MB+ZLIT8[] CLEAR CACHE           'WCTRL/CLRCH.VA_WB,MSRC/@1,SPW/MLONG,ALU/A+B+CI,MUX/M.S,ROT/ZLIT8,LIT/LITRL,LITRL/@2''
;4864   VA_M[]_MB-1                             'WCTRL/VA_WB,MSRC/@1,SPW/MLONG,MUX/M.R1,ALU/A-B-CI,ALUCI/ONE,RSRC/ZERO''
;4865   VA_M[]_MB-ZLIT0[]                       'WCTRL/VA_WB,MSRC/@1,SPW/MLONG,LIT/LITRL,LITRL/@2,ROT/ZLIT0,MUX/M.S,ALU/A-B-CI''
;4866   VA_M[]_MB.AND.R[]                       'WCTRL/VA_WB,MSRC/@1,SPW/MLONG,RSRC/@2,MUX/M.R1,ALU/AND''
;4867   VA_M[]_Q                                'WCTRL/VA_WB,MSRC/@1,SPW/MLONG,RSRC/ZERO,MUX/R.Q,ALU/OR''
;4868   VA_M[]_R[]                              'WCTRL/VA_WB,MSRC/@1,RSRC/@2,ROT/ZERO,MUX/R.S,ALU/OR,SPW/MLONG''
;4869   VA_M[]_R[]+CONX(4)                      'WCTRL/VA_WB,MSRC/@1,RSRC/@2,VSIZE/1,DTYPE/LONG,ROT/CONX.SIZ,MUX/R.S,ALU/A+B+CI,SPW/MLONG''
;4870   VA_M[]_R[]-1                            'WCTRL/VA_WB,MSRC/@1,RSRC/@2,SPW/MLONG,ALU/A+B+CI,MUX/R.S,ROT/MINUS1''
;4871   VA_M[]_R[].OR.Q                         'WCTRL/VA_WB,SPW/MLONG,MSRC/@1,ALU/OR,MUX/R.Q,RSRC/@2''
;4872   VA_PC+SEXT(XB)+2 PC_PC+2                'WCTRL/VA_PC+I+W.PC_PC+I,MSRC/XB.PC_PC+I,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/OR,
;4873                                                                    VSIZE/1,DTYPE/WORD,ISTRM/ISIZE_DSIZE''
;4874   VA_PC+SEXT(XB)+I PC_PC+I                'WCTRL/VA_PC+I+W.PC_PC+I,MSRC/XB.PC_PC+I,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/OR''
;4875   VA_Q                                    'WCTRL/VA_WB,ALU/OR,MUX/R.Q,RSRC/ZERO''
;4876   VA_Q Q_D                                'WCTRL/VA_WB,ALPCTL/WX_Q.Q_D''
;4877   VA_Q_D_D+ZLIT8[]                        'WCTRL/VA_WB,DQ1/Q_D_WX,ROT/ZLIT8,LIT/LITRL,LITRL/@1,MUX/D.S,ALU/A+B+CI''
;4878   VA_Q_D_R[]                              'WCTRL/VA_WB,DQ1/Q_D_WX,RSRC/@1,ROT/ZERO,MUX/R.S,ALU/OR''
;4879   VA_Q_M[]+R[]                            'WCTRL/VA_WB,MSRC/@1,RSRC/@2,MUX/M.R1,ALU/A+B+CI,DQ1/Q_WX''
;4880   VA_RNUM_R[]                             'WCTRL/VA_WB,RSRC/@1,ROT/ZERO,MUX/R.S,ALU/OR,MSRC/RNUM_WBUS''
;4881   VA_RNUM_R[] CLOBBER Q                   'WCTRL/VA_WB,RSRC/@1,ALPCTL/WX_R.Q_M,MSRC/RNUM_WBUS''
;4882   VA_R[]                                  'WCTRL/VA_WB,ALU/OR,MUX/R.S,RSRC/@1,ROT/ZERO''
;4883   VA_R[] Q_D                              'WCTRL/VA_WB,RSRC/@1,ALPCTL/WX_R.Q_D''
;4884   VA_R[] Q_M[]                            'WCTRL/VA_WB,RSRC/@1,ALPCTL/WX_R.Q_M,MSRC/@2''
;4885   VA_R[]+(M[].ASL.P)                      'WCTRL/VA_WB,RSRC/@1,MSRC/@2,ROT/ASL.M.P,MUX/R.S,ALU/A+B+CI''
;4886   VA_R[]+1                                'WCTRL/VA_WB,ALU/A-B-CI,MUX/R.S,RSRC/@1,ROT/MINUS1''
;4887   VA_R[]+CONX(4)                          'WCTRL/VA_WB,ALU/A+B+CI,MUX/R.S,RSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG''
;4888   VA_R[]+CONX.SIZ                         'WCTRL/VA_WB,ALU/A+B+CI,MUX/R.S,RSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/IDEP''
;4889   VA_R[]+M[].PTX                          'WCTRL/VA_WB,ALU/A+B+CI,MUX/R.S,RSRC/@1,ROT/XZ.PTX,MSRC/@2''
;4890   VA_R[]-CONX(1)                          'WCTRL/VA_WB,RSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE,MUX/R.S,ALU/A-B-CI''
;4891   VA_R[]-CONX(2)                          'WCTRL/VA_WB,RSRC/@1,ALU/A-B-CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD''
;4892   VA_R[]-CONX(4)                          'WCTRL/VA_WB,RSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG,MUX/R.S,ALU/A-B-CI''
;4893   VA_R[]-CONX.SIZ                         'WCTRL/VA_WB,RSRC/@1,ALU/A-B-CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/IDEP''
;4894   VA_R[]-M[]                              'WCTRL/VA_WB,ALU/B-A-CI,MUX/M.R1,MSRC/@2,RSRC/@1''
;4895   VA_R[].ASL.SIZE                         'WCTRL/VA_WB,ALPCTL/WX_S,RSRC/@1,ROT/ASL.R.SIZ''
;4896   VA_R[].SIZ_RB-CONX(2)                   'WCTRL/VA_WB,RSRC/@1,SPW/RSIZE,ROT/CONX.SIZ,DTYPE/WORD,VSIZE/1,MUX/R.S,ALU/A-B-CI''
;4897   VA_R[].SIZ_RB-D                         'WCTRL/VA_WB,RSRC/@1,SPW/RSIZE,MUX/D.R1,ALU/B-A-CI''
;4898   VA_R[].XOR.Q                            'WCTRL/VA_WB,ALU/XOR,MUX/R.Q,RSRC/@1''
;4899   VA_R[]_D-CONX(4)                        'WCTRL/VA_WB,RSRC/@1,VSIZE/1,DTYPE/LONG,ROT/CONX.SIZ,MUX/D.S,ALU/A-B-CI,SPW/RLONG''
```

```
;4900    VA_R[]_M[]                      'WCTRL/VA_WB,SPW/RLONG,RSRC/@1,ALU/OR,MUX/M.S,MSRC/@2,ROT/ZERO''
;4901    VA_R[]_M[]+CONX(1)              'WCTRL/VA_WB,SPW/RLONG,RSRC/@1,ALU/A+B+CI,MUX/M.S,MSRC/@2,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4902    VA_R[]_M[]+CONX(2)              'WCTRL/VA_WB,SPW/RLONG,RSRC/@1,ALU/A+B+CI,MUX/M.S,MSRC/@2,ROT/CONX.SIZ,VSIZE/1,DTYPE/WORD''
;4903    VA_R[]_M[]+CONX(4)              'WCTRL/VA_WB,SPW/RLONG,RSRC/@1,ALU/A+B+CI,MUX/M.S,MSRC/@2,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG''
;4904    VA_R[]_M[]+RB                   'WCTRL/VA_WB,RSRC/@1,SPW/RLONG,MSRC/@2,MUX/M.R1,ALU/A+B+CI''
;4905    VA_R[]_M[]-CONX(4)              'WCTRL/VA_WB,RSRC/@1,SPW/RLONG,MSRC/@2,ROT/CONX.SIZ,DTYPE/LONG,MUX/M.S,ALU/A-B-CI,VSIZE/1''
;4906    VA_R[]_M[]-CONX.SIZ             'WCTRL/VA_WB,RSRC/@1,SPW/RLONG,MSRC/@2,ROT/CONX.SIZ,MUX/M.S,ALU/A-B-CI,VSIZE/1,DTYPE/IDEP''
;4907    VA_R[]_Q Q_M[]                  'WCTRL/VA_WB,SPW/RLONG,RSRC/@1,ALPCTL/WX_Q.Q_M,MSRC/@2''
;4908    VA_R[]_RB+T                     'WCTRL/VA_WB,RSRC/@1,SPW/RLONG,ROT/MINUST,MUX/R.S,ALU/A-B-CI''
;4909    VA_R[]_RB+CONX(1)               'WCTRL/VA_WB,SPW/RLONG,RSRC/@1,ALU/A+B+CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4910    VA_R[]_RB+CONX(4)               'WCTRL/VA_WB,RSRC/@1,ROT/CONX.SIZ,MUX/R.S,ALU/A+B+CI,SPW/RLONG,ALUCI/ZERO,VSIZE/1,
;4911                                                DTYPE/LONG''
;4912    VA_R[]_RB+CONX.SIZ              'WCTRL/VA_WB,RSRC/@1,SPW/RLONG,ALU/A+B+CI,MUX/R.S,ROT/CONX.SIZ''
;4913    VA_R[]_RB-CONX(1)               'WCTRL/VA_WB,RSRC/@1,SPW/RLONG,ALU/A-B-CI,MUX/R.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4914    VA_R[]_RB-CONX(2)               'WCTRL/VA_WB,RSRC/@1,SPW/RLONG,ROT/CONX.SIZ,MUX/R.S,ALU/A-B-CI,VSIZE/1,DTYPE/WORD''
;4915    VA_R[]_RB-CONX(4)               'WCTRL/VA_WB,RSRC/@1,ROT/CONX.SIZ,MUX/R.S,ALU/A-B-CI,SPW/RLONG,ALUCI/ZERO,VSIZE/1,
;4916                                                DTYPE/LONG''
;4917    VA_R[]_RB-CONX.SIZ              'WCTRL/VA_WB,RSRC/@1,ROT/CONX.SIZ,MUX/R.S,ALU/A-B-CI,SPW/RLONG,ALUCI/ZERO,VSIZE/1,
;4918                                                DTYPE/IDEP''
;4919    VA_SEXT(XB)+R[] PC_PC+2         'WCTRL/VA_WB,MSRC/XB.PC_PC+I,RSRC/@1,MUX/XM.R,ALUXM/SIGN,ALU/A+B+CI,
;4920                                                VSIZE/1,DTYPE/WORD,ISTRM/ISIZE_DSIZE''
;4921    VA_SEXT(XB)+R[] PC_PC+I         'WCTRL/VA_WB,MSRC/XB.PC_PC+I,RSRC/@1,MUX/XM.R,ALUXM/SIGN,ALU/A+B+CI''
;4922    VA_VA+4                         'WCTRL/VA_VA+4''
;4923    VA_VA+4 CLEAR CACHE             'WCTRL/CLRCH.VA_WB,LITRL/4,ROT/ZLIT0,LIT/LITRL,MSRC/VA,MUX/M.S,ALU/A+B+CI''
;4924    VA_XB PC_PC+4                   'WCTRL/VA_WB,MSRC/XB.PC_PC+I,RSRC/ZERO,MUX/M.R1,ALU/OR,VSIZE/1,DTYPE/LONG,ISTRM/ISIZE_DSIZE''
;4925    VA_XB+R[] PC_PC+4               'WCTRL/VA_WB,MSRC/XB.PC_PC+I,RSRC/@1,MUX/M.R1,ALU/A+B+CI,DTYPE/LONG,
;4926                                                ISTRM/ISIZE_DSIZE,VSIZE/1''
;4927    VA_XB.XOR.R[] PC_PC+4           'WCTRL/VA_WB,ALU/XOR,MUX/XM.R,ALUXM/ZERO,MSRC/XB.PC_PC+I,RSRC/@1,ISTRM/ISIZE_DSIZE,
;4928                                                VSIZE/1,DTYPE/LONG''
;4929    VA_ZEXT(M[])                    'WCTRL/VA_WB,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALUXM/ZERO,ALU/OR''
;4930    VA_ZEXT(M[])+R[]                'WCTRL/VA_WB,ALU/A+B+CI,MUX/XM.R,MSRC/@1,RSRC/@2,ALUXM/ZERO''
;4931    VA_ZEXT(M[])-R[]                'WCTRL/VA_WB,ALU/A-B-CI,MUX/XM.R,MSRC/@1,RSRC/@2,ALUXM/ZERO''
;4932    VA_ZEXT(XB) PC_PC+2             'WCTRL/VA_WB,MSRC/XB.PC_PC+I,RSRC/ZERO,MUX/XM.R,ALUXM/ZERO,ALU/OR,
;4933                                                ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/WORD''
;4934    VA_ZEXT(XB).XOR.(R[].RR.8) PC_PC+1      ''WCTRL/VA_WB,ALU/XOR,MUX/XM.S,MSRC/XB.PC_PC+I,ROT/RR.RR.SIZ,RSRC/@1,
;4935                                                ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/BYTE''
;4936    VA_ZEXT(XB)+R[] PC_PC+1         'WCTRL/VA_WB,MSRC/XB.PC_PC+I,RSRC/@1,MUX/XM.R,ALUXM/ZERO,ALU/A+B+CI,
;4937                                                ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/BYTE''
;4938    VA_ZEXT(XB)-R[] PC_PC+1         'WCTRL/VA_WB,ALU/A-B-CI,MUX/XM.R,MSRC/XB.PC_PC+I,RSRC/@1,ALUXM/ZERO,ISTRM/ISIZE_DSIZE,
;4939                                                VSIZE/1,DTYPE/BYTE''
;4940    VA_ZEXT(XB)-R[] PC_PC+2         'WCTRL/VA_WB,ALU/A-B-CI,MUX/XM.R,MSRC/XB.PC_PC+I,RSRC/@1,ALUXM/ZERO,ISTRM/ISIZE_DSIZE,
;4941                                                VSIZE/1,DTYPE/WORD''
;4942    VA_ZLIT0[]                      'WCTRL/VA_WB,ALPCTL/WX_S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;4943
;4944    WB_(D+CONX(1)).SR.1             ''ALU/A+B+CI.SR,MUX/D.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4945    WB_(M[] R[]).RR.4               ''ALPCTL/WX_S,MSRC/@1,RSRC/@2,ROT/RR.MR.4''
;4946    WB_(M[]+ALKC).BCD               ''ALU/A+B+CI.BCD,ALUCI/ALKC,VSIZE/1,DTYPE/LONG,MUX/M.R1,MSRC/@1,RSRC/ZERO''
;4947    WB_(M[]+R[]).BCD                ''MSRC/@1,RSRC/@2,ALU/A+B+CI.BCD,MUX/M.R1,VSIZE/1,DTYPE/LONG''
;4948    WB_(M[]-R[]).BCD                ''VSIZE/1,DTYPE/LONG,RSRC/@2,MSRC/@1,MUX/M.R1,ALU/A-B-CI.BCD''
;4949    WB_(M[]-R[]-ALKC).BCD           ''VSIZE/1,DTYPE/LONG,RSRC/@2,MSRC/@1,MUX/M.R1,ALU/A-B-CI.BCD,ALUCI/ALKC''
;4950    WB_(M[].AND.ZLIT0[]).SR         ''MSRC/@1,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@2,ALU/AND.SR''
;4951    WB_(R[] M[]).RL.P               ''RSRC/@1,MSRC/@2,ROT/RL.RM.P,ALPCTL/WX_S''
;4952    WB_(R[] M[]).RL.PS              ''RSRC/@1,MSRC/@2,ROT/RL.RM.PS,ALPCTL/WX_S''
;4953    WB_(R[]+Q).SL.1                 ''MUX/R.Q,RSRC/@1,ALU/A+B+CI.SL,ALUSHF/ZERO''
;4954    WB_.NOT.(M[].RR.16)             ''ALPCTL/WX_.NOT.S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,MSRC/@1''
```

```
;4955   WB_.NOT.(M[].RR.P)              ''ALPCTL/WX_.NOT.S,ROT/RR.MM.P,MSRC/@1''
;4956   WB_.NOT.Q                      ''ALU/A-B-CI,MUX/R.Q,RSRC/ZERO,ALUCI/ONE''
;4957   WB_0-CONX(1)                   ''ALU/A-B-CI,MUX/Z.S,ROT/CONX.SIZ,VSIZE/1,DTYPE/BYTE''
;4958   WB_ATCR                        ''CCMISC/WB_ATCR.CCBR_SIGND''
;4959   WB_D                           ''MUX/D.R1,RSRC/ZERO,ALU/OR''
;4960   WB_D+ALKC                      ''MUX/D.R1,RSRC/ZERO,ALU/A+B+CI,ALUCI/ALKC''
;4961   WB_D+ZLIT0[]                   ''ALU/A+B+CI,MUX/D.S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;4962   WB_D+ZLIT16[]                  ''ALU/A+B+CI,MUX/D.S,ROT/ZLIT16,LIT/LITRL,LITRL/@1''
;4963   WB_D-OLIT24[]                  ''MUX/D.S,ROT/OLIT24,LIT/LITRL,LITRL/@1,ALU/A-B-CI''
;4964   WB_D-PL                        ''MUX/D.S,ROT/PL,ALU/A-B-CI''
;4965   WB_D-Q                         ''ALU/A-B-CI,ALUCI/ZERO,MUX/D.Q1''
;4966   WB_D-R[]                       ''RSRC/@1,MUX/D.R1,ALU/A-B-CI''
;4967   WB_D-ZLIT0[]                   ''LIT/LITRL,LITRL/@1,ROT/ZLIT0,MUX/D.S,ALU/A-B-CI''
;4968   WB_D.AND.ZLIT0[]               ''ALU/AND,MUX/D.S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;4969   WB_D.AND.ZLIT16[]              ''LIT/LITRL,LITRL/@1,ROT/ZLIT16,MUX/D.S,ALU/AND''
;4970   WB_D.AND.ZLIT24[]              ''ALU/AND,MUX/D.S,ROT/ZLIT24,LIT/LITRL,LITRL/@1''
;4971   WB_D.ANDNOT.ZLIT0[]            ''LIT/LITRL,LITRL/@1,ROT/ZLIT0,MUX/D.S,ALU/ANDNOT''
;4972   WB_D.XOR.R[]                   ''ALU/XOR,MUX/D.R1,RSRC/@1''
;4973   WB_D.XOR.ZLIT0[]               ''ALU/XOR,MUX/D.S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;4974   WB_D.XOR.ZLIT16[]              ''ALU/XOR,MUX/D.S,ROT/ZLIT16,LIT/LITRL,LITRL/@1''
;4975   WB_D.XOR.ZLIT24[]              ''ALU/XOR,MUX/D.S,ROT/ZLIT24,LIT/LITRL,LITRL/@1''
;4976   WB_EXP(M[])                    ''MSRC/@1,ROT/GETEXP,ALPCTL/WX_S''
;4977   WB_EXP(M[]) Q_ZEXT(MB)         ''MSRC/@1,ROT/GETEXP,ALPCTL/WX_S.Q_XM,ALUXM/ZERO''
;4978   WB_FPA                         ''FPA/WBUS_FPA''
;4979   WB_M[]                         ''ALU/OR,MUX/M.S,ROT/ZERO,MSRC/@1''
;4980   WB_M[] PL_MB.MSS               ''MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/OR,ROT/PL_MSS''
;4981   WB_M[]+CONX(1)                 ''ALU/A+B+CI,MUX/M.S,MSRC/@1,ROT/CONX.SIZ,DTYPE/BYTE,VSIZE/1''
;4982   WB_M[]+PSLC                    ''MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/A+B+CI,ALUCI/PSLC''
;4983   WB_M[]+Q                       ''MUX/M.Q1,MSRC/@1,ALU/A+B+CI''
;4984   WB_M[]+Q+PSLC                  ''MSRC/@1,MUX/M.Q1,ALU/A+B+CI,ALUCI/PSLC''
;4985   WB_M[]+R[]+ALKC                ''ALU/A+B+CI,MUX/M.R1,MSRC/@1,RSRC/@2,ALUCI/ALKC''
;4986   WB_M[]+R[]+PSLC                ''ALU/A+B+CI,ALUCI/PSLC,MUX/M.R1,MSRC/@1,RSRC/@2''
;4987   WB_M[]+ZLIT0[]                 ''ALU/A+B+CI,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4988   WB_M[]-(MB.CLR2B)              ''ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/CLR2BM''
;4989   WB_M[]-(R[].XZ)                ''ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/XZ.RR,RSRC/@2''
;4990   WB_M[]-1                       ''MSRC/@1,ROT/MINUS1,MUX/M.S,ALU/A+B+CI''
;4991   WB_M[]-PL                      ''ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/PL''
;4992   WB_M[]-PSLC                    ''MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/A-B-CI,ALUCI/PSLC''
;4993   WB_M[]-Q                       ''MUX/M.Q1,MSRC/@1,ALU/A-B-CI''
;4994   WB_M[]-Q-PSLC                  ''MSRC/@1,MUX/M.Q1,ALU/A-B-CI,ALUCI/PSLC''
;4995   WB_M[]-R[]                     ''MUX/M.R1,MSRC/@1,RSRC/@2,ALU/A-B-CI''
;4996   WB_M[]-R[]-ALKC                ''MUX/M.R1,MSRC/@1,RSRC/@2,ALU/A-B-CI,ALUCI/ALKC''
;4997   WB_M[]-ZLIT0[]                 ''ALU/A-B-CI,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;4998   WB_M[]-ZLIT8[]                 ''MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT8,MUX/M.S,ALU/A-B-CI''
;4999   WB_M[].AND.CONX(1)             ''ALU/AND,MUX/M.S,MSRC/@1,ROT/CONX.SIZ,DTYPE/BYTE,VSIZE/1''
;5000   WB_M[].AND.CONX(2)             ''ALU/AND,MUX/M.S,MSRC/@1,ROT/CONX.SIZ,DTYPE/WORD,VSIZE/1''
;5001   WB_M[].AND.CONX(4)             ''ALU/AND,MUX/M.S,MSRC/@1,ROT/CONX.SIZ,DTYPE/LONG,VSIZE/1''
;5002   WB_M[].AND.OLIT0[]             ''MSRC/@1,LIT/LITRL,LITRL/@2,ROT/OLIT0,MUX/M.S,ALU/AND''
;5003   WB_M[].AND.Q                   ''MUX/M.Q1,MSRC/@1,ALU/AND''
;5004   WB_M[].AND.R[]                 ''ALU/AND,MUX/M.R1,MSRC/@1,RSRC/@2''
;5005   WB_M[].AND.ZLIT0[]             ''MSRC/@1,LITRL/@2,MUX/M.S,ALU/AND,ROT/ZLIT0,LIT/LITRL''
;5006   WB_M[].AND.ZLIT12[]            ''MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT12,MUX/M.S,ALU/AND''
;5007   WB_M[].AND.ZLIT16[]            ''ALU/AND,MUX/M.S,MSRC/@1,ROT/ZLIT16,LIT/LITRL,LITRL/@2''
;5008   WB_M[].AND.ZLIT20[]            ''ALU/AND,MUX/M.S,MSRC/@1,ROT/ZLIT20,LIT/LITRL,LITRL/@2''
;5009   WB_M[].AND.ZLIT24[]            ''ALU/AND,MUX/M.S,MSRC/@1,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
```

```
;5010    WB_M[].AND.ZLIT4[]                  ''MSRC/@1,LIT/LITRL,LITRL/@2,ROT/ZLIT4,MUX/M.S,ALU/AND''
;5011    WB_M[].AND.ZLIT8[]                  ''ALU/AND,MUX/M.S,MSRC/@1,ROT/ZLIT8,LIT/LITRL,LITRL/@2''
;5012    WB_M[].ANDNOT.ZLIT0[]               ''ALU/ANDNOT,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;5013    WB_M[].ANDNOT.ZLIT24[]              ''ALU/ANDNOT,MUX/M.S,MSRC/@1,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;5014    WB_M[].ASR.-P                       ''MSRC/@1,ROT/ASR.M.-P,ALPCTL/WX_S''
;5015    WB_M[].ASR.P                        ''ALPCTL/WX_S,ROT/ASR.M.P,MSRC/@1''
;5016    WB_M[].BCDSWP                       ''MSRC/@1,ROT/BCDSWP,MUX/R.S,RSRC/ZERO,ALU/OR''
;5017    WB_M[].CLR2B                        ''ALPCTL/WX_S,ROT/CLR2BM,MSRC/@1''
;5018    WB_M[].OR.R[]                       ''MSRC/@1,MUX/M.R1,ALU/OR,RSRC/@2''
;5019    WB_M[].RR.16                        ''ALPCTL/WX_S,ROT/RR.MM.SIZ,MSRC/@1,VSIZE/1,DTYPE/WORD''
;5020    WB_M[].RR.24                        ''ALPCTL/WX_S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/LONG,MSRC/@1''
;5021    WB_M[].RR.4                         ''ALPCTL/WX_S,ROT/RR.MR.4,MSRC/@1,RSRC/@1''
;5022    WB_M[].RR.8                         ''ALPCTL/WX_S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/BYTE,MSRC/@1''
;5023    WB_M[].RR.P                         ''ALPCTL/WX_S,ROT/RR.MM.P,MSRC/@1''
;5024    WB_M[].VPN-R[]                      ''ALU/B-A-CI,MUX/R.S,RSRC/@2,ROT/XZ.VPN,MSRC/@1''
;5025    WB_M[].XOR.Q                        ''MSRC/@1,MUX/M.Q1,ALU/XOR''
;5026    WB_M[].XOR.Q Q_(Q.SL.1).OR.1        ''ALU/XOR,MUX/M.Q2,MSRC/@1,DQ2/SQL,ALUSHF/ONE''
;5027    WB_M[].XOR.Q Q_Q.RL.1               ''ALU/XOR,MUX/M.Q2,MSRC/@1,DQ2/SQL,ALUSHF/ROT''
;5028    WB_M[].XOR.R[]                      ''ALU/XOR,MUX/M.R1,MSRC/@1,RSRC/@2''
;5029    WB_M[].XOR.ZLIT0[]                  ''ALU/XOR,MUX/M.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;5030    WB_M[].XOR.ZLIT8[]                  ''ALU/XOR,MUX/M.S,MSRC/@1,ROT/ZLIT8,LIT/LITRL,LITRL/@2''
;5031    WB_M[].XZ.MM                        ''MSRC/@1,ALPCTL/WX_S,ROT/XZ.MM''
;5032    WB_NOT(M[].ASR.-P)                  ''MSRC/@1,ROT/ASR.M.-P,ALPCTL/WX_.NOT.S''
;5033    WB_PL*2                             ''ROT/PL,RSRC/ZERO,MUX/R.S,ALU/A+B+CI.SL''
;5034    WB_PSL                              ''CCPSL/WB_PSL.CCBR_SIGND''
;5035    WB_Q                                ''ALU/OR,MUX/R.Q,RSRC/ZERO''
;5036    WB_Q Q_M[]                          ''MSRC/@1,ALPCTL/WX_Q.Q_M''
;5037    WB_Q-D                              ''MUX/D.Q1,ALU/B-A-CI,ALUCI/ZERO''
;5038    WB_Q-M[]                            ''MUX/M.Q1,MSRC/@1,ALU/B-A-CI''
;5039    WB_Q-R[]                            ''RSRC/@1,MUX/R.Q,ALU/B-A-CI''
;5040    WB_Q_M[]                            ''MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/OR,DQ1/Q_WX''
;5041    WB_RNUM                             ''MSRC/WBUS_RNUM''
;5042    WB_R[]                              ''RSRC/@1,MUX/R.S,ROT/ZERO,ALU/OR''
;5043    WB_R[]+0                            ''ALU/A+B+CI,MUX/R.S,RSRC/@1,ROT/ZERO''
;5044    WB_R[]+1                            ''ALU/A-B-CI,MUX/R.S,RSRC/@1,ROT/MINUS1''
;5045    WB_R[]-0                            ''RSRC/@1,ROT/ZERO,MUX/R.S,ALU/A-B-CI''
;5046    WB_R[]-CONX(4)                      ''RSRC/@1,ROT/CONX.SIZ,VSIZE/1,DTYPE/LONG,MUX/R.S,ALU/A-B-CI''
;5047    WB_R[]-D                            ''RSRC/@1,MUX/D.R1,ALU/B-A-CI''
;5048    WB_R[]-M[]                          ''RSRC/@1,MSRC/@2,MUX/M.R1,ALU/B-A-CI,ALUCI/ZERO''
;5049    WB_R[]-Q                            ''RSRC/@1,MUX/R.Q,ALU/A-B-CI''
;5050    WB_R[]-XB PC_PC+4                   ''MSRC/XB.PC_PC+1,RSRC/@1,MUX/M.R1,ALU/B-A-CI,VSIZE/1,
;5051                                                DTYPE/LONG,ISTRM/ISIZE_DSIZE''
;5052    WB_R[]-ZEXT(M[])                    ''RSRC/@1,MSRC/@2,MUX/XM.R,ALUXM/ZERO,ALU/B-A-CI''
;5053    WB_R[]-ZEXT(XB) PC_PC+1             ''MSRC/XB.PC_PC+1,RSRC/@1,ALUXM/ZERO,MUX/XM.R,ALU/B-A-CI,VSIZE/1,
;5054                                                DTYPE/BYTE,ISTRM/ISIZE_DSIZE''
;5055    WB_R[].AND.Q                        ''RSRC/@1,MUX/R.Q,ALU/AND''
;5056    WB_R[].NOTAND.Q                     ''ALU/NOTAND,MUX/R.Q,RSRC/@1''
;5057    WB_R[].OR.(M[].ASR.-P)              ''RSRC/@1,MUX/R.S,ALU/OR,MSRC/@2,ROT/ASR.M.-P''
;5058    WB_R[].RR.16                        ''ALPCTL/WX_S,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/WORD,RSRC/@1''
;5059    WB_R[].RR.24                        ''ALPCTL/WX_S,RSRC/@1,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/LONG''
;5060    WB_R[].RR.8                         ''ALPCTL/WX_S,ROT/RR.RR.SIZ,VSIZE/1,DTYPE/BYTE,RSRC/@1''
;5061    WB_R[].RR.SIZ                       ''RSRC/@1,ROT/RR.RR.SIZ,ALPCTL/WX_S''
;5062    WB_R[].SR.5(IF MDR IS 0)            ''RSRC/@1,MSRC/MDR,ROT/RR.MR.4,MUX/M.S,ALU/A+B+CI.SR''
;5063    WB_R[].XOR.Q                        ''ALU/XOR,MUX/R.Q,RSRC/@1''
;5064    WB_SEXT(M[])-Q                      ''MSRC/@1,ALUXM/SIGN,MUX/XM.Q,ALU/A-B-CI''
```

```
;5065   WB_SEXT(M[]).ANDNOT.ZLIT0[]      ''ALU/ANDNOT,MUX/XM.S,MSRC/@1,ROT/ZLIT0,LIT/LITRL,LITRL/@2''
;5066   WB_SEXT(XB) PC_PC+2               ''MSRC/XB.PC_PC+1,RSRC/ZERO,ALUXM/SIGN,ALU/OR,ISTRM/ISIZE_DSIZE,VSIZE/1,DTYPE/WORD''
;5067   WB_SEXT(XB)-R[] PC_PC+1           ''MSRC/XB.PC_PC+1,RSRC/@1,ALU/A-B-CI,MUX/XM.R,ALUXM/SIGN,VSIZE/1,
;5068                                              DTYPE/BYTE,ISTRM/ISIZE_DSIZE''
;5069   WB_SEXT(XB).XOR.MINUS1 PC_PC+1    ''MSRC/XB.PC_PC+1,ROT/MINUS1,MUX/XM.S,ALUXM/SIGN,ALU/XOR,DTYPE/BYTE,VSIZE/1,
;5070                                              ISTRM/ISIZE_DSIZE''
;5071   WB_SEXT(XB).XOR.MINUS1 PC_PC+2    ''MSRC/XB.PC_PC+1,ROT/MINUS1,MUX/XM.S,ALUXM/SIGN,ALU/XOR,DTYPE/WORD,VSIZE/1,
;5072                                              ISTRM/ISIZE_DSIZE''
;5073   WB_XB-R[] PC_PC+4                 ''MSRC/XB.PC_PC+1,RSRC/@1,MUX/M.R1,ALU/A-B-CI,VSIZE/1,
;5074                                              DTYPE/LONG,ISTRM/ISIZE_DSIZE''
;5075   WB_ZEXT(M[])                      ''MSRC/@1,MUX/XM.S,ROT/ZERO,ALU/OR,ALUXM/ZERO''
;5076   WB_ZEXT(M[])-R[]                  ''MSRC/@1,RSRC/@2,MUX/XM.R,ALUXM/ZERO,ALU/A-B-CI''
;5077   WB_ZEXT(M[]).AND.R[]              ''MSRC/@1,RSRC/@2,MUX/XM.R,ALUXM/ZERO,ALU/AND''
;5078   WB_ZEXT(XB) PC_PC+1               ''ALU/OR,ALUXM/ZERO,MUX/XM.R,MSRC/XB.PC_PC+1,RSRC/ZERO,VSIZE/1,DTYPE/BYTE,ISTRM/ISIZE_DSIZE''
;5079   WB_ZEXT(XB)-(R[].RR.16) PC_PC+2   ''MSRC/XB.PC_PC+1,RSRC/@1,MUX/XM.S,ROT/RR.RR.SIZ,ALU/A-B-CI,VSIZE/1,
;5080                                              DTYPE/WORD,ISTRM/ISIZE_DSIZE''
;5081   WB_ZEXT(XB)-R[] PC_PC+1           ''MSRC/XB.PC_PC+1,RSRC/@1,ALUXM/ZERO,MUX/XM.R,ALU/A-B-CI,VSIZE/1,
;5082                                              DTYPE/BYTE,ISTRM/ISIZE_DSIZE''
;5083   WB_ZEXT(XB)-R[] PC_PC+2           ''MSRC/XB.PC_PC+1,RSRC/@1,ALUXM/ZERO,MUX/XM.R,ALU/A-B-CI,VSIZE/1,
;5084                                              DTYPE/WORD,ISTRM/ISIZE_DSIZE''
;5085   WB_ZLIT0[]-M[]                    ''MSRC/@2,ALU/B-A-CI,MUX/M.S,ROT/ZLIT0,LIT/LITRL,LITRL/@1''
;5086
;5087   WDR_UNROT(M[].OR.ZLIT24[])        ''WCTRL/WDR_WB.UR,ALU/OR,MUX/M.S,MSRC/@1,ROT/ZLIT24,LIT/LITRL,LITRL/@2''
;5088   WDR_UNROT(R[])                    ''WCTRL/WDR_WB.UR,ALU/OR,MUX/R.S,RSRC/@1,ROT/ZERO''
;5089   WX_CONX.SIZ                       ''ALUOD/OR.OD,MUX/Z.S,ROT/CONX.SIZ''
```

```
;5090    .TOC "  Branching Macros"
;5091
;5092    (M[TEMP3]-SL)BYTE RANGE CHECK?  "BUT/SRKSTA,MSRC/@1,MUX/M.S,ALU/A-B-CI,ROT/SL"
;5093    (PL+SL).GT.32?                  "BUT/SRKSTA,ROTSRK/VIELD.000"
;5094
;5095    ABSVAL M[]<7-0>?                "BUT/SRKSTA,ROT/MINUS1,MSRC/@1,MUX/M.S,ALU/AND"
;5096    ACLO FPLOCK?                    "BUT/FPS3"
;5097    ADD1(FLAG0)?                    "BUT/FLAG0"
;5098    ADD2(FLAG1) ADD1(FLAG0)?        "BUT/FLAG1T00"
;5099    ALKC?                           "BUT/WBUS31T030,ALPCTL/WB_ALUF"
;5100    ALLOW INT?                      "BUT/CCBR1.INT-TS"
;5101    ALUS?                           "BUT/CCBR,CC/NOP.CCBR_ALUS"
;5102    ALUS_UNSGN OLDALUS?             "BUT/CCBR,CCMISC/ALUS_UNSGN.CCBR_ALUS"
;5103    ASCII SIGN(M[])?                "BUT/SRKSTA,MSRC/@1,RSRC/ZERO,ALU/OR,MUX/M.R1,ROTSRK/ASCIISIGN.073"
;5104
;5105    BCD CHECK?                      "BUT/BCDCHK"
;5106    BCD CHECK M[]?                  "BUT/BCDCHK,MSRC/@1,ALU/A+B+CI.BCD,MUX/R.S,RSRC/ZERO,ROT/CVTNP"
;5107    BCD SIGN M[]?                   "BUT/SRKSTA,ROT/BCDSWP,MSRC/@1"
;5108    BCD SIGN.ZERO?                  "BUT/CCBR,CC/NOP.CCBR_ALUS"
;5109    BCD SIGN.ZERO(DEF)?             "BUT/CCBR"
;5110
;5111    BINARY LOAD?                    "BUT/CCBR,CC/NOP.CCBR_ALUS"
;5112    BOOT(FLAG MMNOINT)?             "BUT/MM.NOINT"
;5113    BRA ON ADD?                     "BUT/BRA.ON.ADD"
;5114
;5115    CCOP1 SIGND?                    "BUT/CCBR,CC/CCOP1.CCBR_SIGND"
;5116    CCOP2 SIGND CMP .NOT.IR0?       "BUT/CCBR1.CCBR0.IR0,CC7CCOP2.CCBR_SIGND"
;5117    CC_ZLIT0[] ALUS?                "BUT/CCBR,CCPSL/CC_WB.CCBR_ALUS,LIT/LITRL,LITRL/@1,ROT/ZLIT0,ALPCTL/WX_S"
;5118    CHECK INTERRUPTS?               "BUT/CCBR1.INT-TS,VSIZE/1,DTYPE/LONG"
;5119    CMP SIGNS?                      "BUT/CCBR,CCMISC/NOP.CCBR_CSIGNS"
;5120    COUNT OR INT TIMER?             "BUT/CCBR1.INT-TS"
;5121
;5122    DIVIDEND SIGN?                  "MSRC/TEMP1,BUT/FR0.FLTZ"
;5123    DBZ STEPC?                      "BUT/DBZ.SC"
;5124    DSIZE?                          "BUT/DSIZE"
;5125
;5126    EMODH(FLAG4)?                   "BUT/MM.NOINT"
;5127    EXPONENT RANGE?                 "BUT/SRKSTA"
;5128
;5129    FLAG0?                          "BUT/FLAG0"
;5130    FLAG1 (FLAG2.XOR.FLAG3)?        "BUT/F1.XOR23"
;5131    FLAG1?                          "BUT/FLAG1"
;5132    FLAG2?                          "BUT/FLAG2"
;5133    FLAG3?                          "BUT/FLAG3"
;5134    FLAG4?                          "BUT/MM.NOINT"
;5135    FLAG<1-0>?                      "BUT/FLAG1T00"
;5136    FLAG<2-0>?                      "BUT/FLAG2T00"
;5137    FPA PRESENT?                    "BUT/NO.FPA"
;5138    FPA(FLAG0)?                     "BUT/FLAG0"
;5139    FPD?                            "BUT/FPD"
;5140    FPS1?                           "BUT/FPS1"
;5141    FPS2?                           "BUT/FPS2"
;5142    FPS3?                           "BUT/FPS3"
;5143    FR0.FLTZ?                       "BUT/FR0.FLTZ"
;5144
```

```
;5145   GFLOAT(FLAG4)?                    ''BUT/MM.NOINT''
;5146
;5147   INTPEND OR TIMER?                 ''BUT/INT-TIMSERV''
;5148   IP.TS(SIGND CMP)?                 ''CC/NOP.CCBR_SIGND,BUT/CCBR1.INT-TS''
;5149   IP.TS?                            ''BUT/CCBR1.INT-TS,CCMISC/NOP.CCBR_BRATST''
;5150   IR<2>?                            ''BUT/IR2''
;5151   IR<5>?                            ''BUT/IR5''
;5152   IR<2-0>?                          ''BUT/IR.2TO0''
;5153
;5154   LOD BRA?                          ''BUT/LOD.BRA''
;5155   LOD INC BRA?                      ''BUT/LOD.INC.BRA''
;5156
;5157   MDR_GPR.R RNUM.EQ.7?              ''BUT/SPASTA,WCTRL/MDR_WB,RSRC/GPR.R,ROT/ZERO,MUX/R.S,ALU/OR''
;5158   MDR_ZEXT(OSR) BRATST?             ''BUT/CCBR,CCPSL/MDR_OSR.CCBR_BRATST''
;5159
;5160   MICRO VECTOR?                     ''BUT/UVCTR,CLKX/XTND''
;5161
;5162   MM.ALLOW.INT?                     ''BUT/MM.ALLOW.INT''
;5163   MM.NOINT?                         ''BUT/MM.NOINT''
;5164
;5165   MOPZERO (MUL1 XOR MUL2)?          ''BUT/F1.XOR23''
;5166   MOPZERO(FLAG1)?                   ''BUT/FLAG1''
;5167
;5168   MUL1(FLAG2) XOR MUL2(FLAG3)?      ''BUT/F1.XOR23''
;5169   MUL2(FLAG3)?                      ''BUT/FLAG3''
;5170
;5171   M[] BIT24?                        ''BUT/WX.NE.0,ALU/AND,MUX/M.S,MSRC/@1,ROT/ZLIT24,LIT/LITRL,LITRL/1''
;5172   M[].EQ.-1?                        ''BUT/CCBR0.SRKSTA0,CC/NOP.CCBR_SIGND,MSRC/@1,MUX/M.S,ALU/NOTAND,ROT/MINUS1''
;5173   M[]<31-16>.EQ.FFFF?               ''BUT/CCBR,ALPCTL/WX_.NOT.S,ROT/RR.MM.SIZ,VSIZE/1,DTYPE/WORD,MSRC/@1,CC/NOP.CCBR_SIGND''
;5174
;5175   ODD ADDRESS?                      ''BUT/CM.ODD.ADD''
;5176   OTHER UTRAPS?                     ''BUT/BUTXB''
;5177   OPZERO(FLAG3)?                    ''BUT/FLAG3''
;5178   OVER(FLAG2)?                      ''BUT/FLAG2''
;5179
;5180   PL<4-0>.EQ.0?                     ''BUT/SRKSTA''
;5181   PL<4-0>.EQ.0? PL<5>?              ''BUT/SRKSTA,ROTSRK/PL.EQ.0.SIGN.120''
;5182   POPIC(FLAG4)?                     ''BUT/MM.NOINT''
;5183   PC>>31? (PL+SL)>32?               ''BUT/CCBR0.SRKSTA0,CC/NOP.CCBR_ALUS,ROTSRK/VIELD.012''
;5184
;5185   NOT.Q<31>?                        ''BUT/WBUS31TO30,RSRC/ZERO,MUX/R.Q,ALU/A-B-CI.SR,ALUCI/ONE,ALUSHF/ZERO''
;5186
;5187   PROBE READ MODE ON WB?            ''BUT/UVCTR,CLKX/XTND,BUS/PRB.RD.MODE''
;5188   PROBE READ?                       ''BUT/UVCTR,CLKX/XTND,BUS/PRB.RD''
;5189   PROBE WRITE MODE ON WB?           ''BUT/UVCTR,CLKX/XTND,BUS/PRB.WR.MODE''
;5190   PROBE WRITE?                      ''BUT/UVCTR,CLKX/XTND,BUS/PRB.WR''
;5191
;5192   PSL<C>?                           ''BUT/PSLC''
;5193   PSL<IS.CURM>?                     ''BUT/UVCTR,CLKX/XTND,WCTRL/UVCTR_CM.IS''
;5194   PSL<TP>?                          ''BUT/PSLTP''
;5195
;5196   PTE CHECK READ KERNAL?            ''BUT/UVCTR,CLKX/XTND,BUS/PRB.RD.PTE.K''
;5197   PTE CHECK READ?                   ''BUT/UVCTR,CLKX/XTND,BUS/PRB.RD.PTE''
;5198   PTE CHECK WRITE?                  ''BUT/UVCTR,CLKX/XTND,BUS/PRB.WR.PTE''
;5199
```

```
;5200    Q IS NEG?                       'BUT/CCBR,RSRC/ZERO,MUX/R.Q,ALU/OR,CC/NOP.CCBR_SIGND''
;5201    Q IS ZERO?                      'BUT/WX.EQ.0,MUX/R.Q,RSRC/ZERO,ALU/OR,DQ1/Q_WX''
;5202    Q.NOT IS POS?                   'BUT/WBUS31TO30,RSRC/ZERO,MUX/R.Q,ALU/A-B-CI,ALUCI/ONE''
;5203
;5204    R6_RB+CONX(2) RNUM.EQ.7?        'BUT/SPASTA,RSRC/R6,SPW/RLONG,ROT/CONX.SIZ,MUX/R.S,ALU/A+B+CI,VSIZE/1,DTYPE/WORD''
;5205    R7_M[] RNUM.EQ.7?               'BUT/SPASTA,RSRC/R7,SPW/RLONG,MSRC/@1,ALU/OR,MUX/M.S,ROT/ZERO''
;5206    RBS + OR - ?                    'BUT/SPASTA''
;5207    RBSP.EQ.0?                      'BUT/SPASTA''
;5208    READ(FLAG1) FPA(FLAG0)?         'BUT/FLAG1TO0''
;5209    READ(FLAG1)?                    'BUT/FLAG1''
;5210    REG MODE?                       'BUT/REGMODE''
;5211    REGINT(FLAG1) ADD1(FLAG0)?      'BUT/FLAG1TO0''
;5212    REGINT(FLAG1)?                  'BUT/FLAG1''
;5213    REI CHECK?                      'BUT/UVCTR,CLKX/XTND,WCTRL/REICHK,BUS/REICHK''
;5214    RNUM.EQ.7?                      'BUT/SPASTA,RSRC/GPR.R''
;5215    RNUM_R[] IPR CHECK?             'BUT/SPASTA,MSRC/RNUM_WBUS,ALU/OR,MUX/R.S,RSRC/@1,ROT/ZERO''
;5216
;5217    S<3-0>.NE.0?                    'BUT/SRKSTA,ROTSRK/BCDSIGN.063''
;5218    SAMESIGN(FLAG4)?                'BUT/MM.NOINT''
;5219    SET V SIGND CMP?                'BUT/CCBR,CCMISC/SETV.CCBR_SIGND''
;5220    SHIFT SIZE?                     'BUT/SRKSTA,ROTSRK/PL.EQ.0.SIGN.121''
;5221    SIGN(FLAG0)?                    'BUT/FLAG0''
;5222    SIGND CMP .NOT.IR0?             'BUT/CCBR1.CCBR0.IR0,CC/NOP.CCBR_SIGND''
;5223    SIGND CMP DEF?                  'BUT/CCBR''
;5224    SIGND CMP?                      'BUT/CCBR,CC/NOP.CCBR_SIGND''
;5225    SL_D_SEXT(M[]) WBRANGE?         'BUT/SRKSTA,ROT/PL.SL_WB,MSRC/@1,RSRC/ZERO,MUX/XM.R,ALUXM/SIGN,ALU/OR,DQ1/D_WX''
;5226    SL_M[] WBRANGE?                 'BUT/SRKSTA,ROT/PL.SL_WB,MSRC/@1,RSRC/ZERO,MUX/M.R1,ALU/OR''
;5227    STACK FLAG?                     'BUT/STACKFLG''
;5228    STEP(.GE.4? DELBY4              'BUT/BR.SC-4.INT-TS''
;5229    SUB(FLAG1)?                     'BUT/FLAG1''
;5230
;5231    TIMER?                          'BUT/INT-TIMSERV''
;5232
;5233    VA<0>?                          'BUT/WBUS0,MSRC/VA,RSRC/ZERO,MUX/M.R1,ALU/OR''
;5234    VA_R6_RB-CONX(2) RNUM.EQ.7?     'BUT/SPASTA,WCTRL/VA_WB,RSRC/R6,SPW/RLONG,ROT/CONX.SIZ,MUX/R.S ALU/A-B-CI,VSIZE/1,
;5235                                                DTYPE/WORD''
;5236
;5237    WB<0>?                          'BUT/WBUS0''
;5238    WB<1-0>?                        'BUT/WBUS1TO0''
;5239    WB<1-0>.NE.0?                   'BUT/WBUS1TO0.NE.0''
;5240    WX<15-0>.NE.0?                  'BUT/SRKSTA''
;5241    WX<31-16>.NE.0?                 'BUT/SRKSTA''
;5242    WB<31-30>?                      'BUT/WBUS31TO30''
;5243    WB<5-0>?                        'BUT/WBUS5TO0''
;5244    WB_M[]+63 PL_31 WB<7>EQ0?       'BUT/SRKSTA,MSRC/@1,MUX/M.S,ALU/A+B+CI,ROT/OLITO.PL_LIT,LIT/LITRL,LITRL/3F,VSIZE/1,
;5245                                                DTYPE/BYTE''
;5246    WCS DISABLED?                   'BUT/WCSENA''
;5247    WRITE(FLAG1)?                   'BUT/FLAG1''
;5248    WX(SIZ).EQ.0?                   'BUT/CCBR0.SRKSTA0,CC/NOP.CCBR_SIGND''
;5249    WX.EQ.0?                        'BUT/WX.EQ.0''
;5250    WX.NE.0?                        'BUT/WX.NE.0''
;5251
;5252    ZEXT(M[]).EQ.0?                 'BUT/WX.EQ.0,MSRC/@1,MUX/XM.R,ALUXM/ZERO,ALU/OR,RSRC/ZERO''
```

```
;5253    .TOC    "       Ird and Dsize Rom Macros"
;5254
;5255    .NOCREF                 ;SET UP FOR NEVER CREF
;5256    .NOCREF
;5257
;5258    .ICODE
;5259    FPD []{} []{}           "FOP/@1,FPD/<.SHIFT[<.AND[<NEXT/@2>,3F8]>,-3]>,FFOP/@3,FPD.FPA/<.SHIFT[<.AND[<NEXT/@4>,3F8]>,-3]>,
;5260                             VFPD/<.EQL[0,  <.AND[<NEXT/@2>,3C07]>,  <.AND[<NEXT/@4>,3C07]>  ]>"
;5261
;5262    IRD1[]{} []{}           "IOP/@1,IRD1/<.SHIFT[<.AND[<NEXT/@2>,3F8]>,-3]>,IFOP/@3,IRD1.FPA/<.SHIFT[<.AND[<NEXT/@4>,3F8]>,-3]>,
;5263                             VIRD1/<.EQL[0,  <.AND[<NEXT/@2>,3C07]>,  <.AND[<NEXT/@4>,3C07]>  ]>"
;5264
;5265    .OCODE
;5266    CNT0[]{}{} []{}{}       "OOP/ @1,CNT0.REG/    <.AND[<NEXT/@2>,7FF]>,CNT0.MEM/    <.AND[<NEXT/@3>,7FF]>,
;5267                             OFOP/@4,CNT0.FPA.REG/<.AND[<NEXT/@5>,7FF]>,CNT0.FPA.MEM/<.AND[<NEXT/@6>,7FF]>,
;5268                             VCNT0/<.EQL[ 0, <.AND[<NEXT/@2>,3800]> , <.AND[<NEXT/@5>,3800]> ]>"
;5269
;5270    CNT1[]{}{} []{}{}       "1OP/ @1,CNT1.REG/    <.AND[<NEXT/@2>,7FF]>,CNT1.MEM/    <.AND[<NEXT/@3>,7FF]>,
;5271                             1FOP/@4,CNT1.FPA.REG/<.AND[<NEXT/@5>,7FF]>,CNT1.FPA.MEM/<.AND[<NEXT/@6>,7FF]>,
;5272                             VCNT1/<.EQL[ 0, <.AND[<NEXT/@2>,3800]> , <.AND[<NEXT/@5>,3800]> ]>"
;5273
;5274    .CCODE
;5275    REG [] [] []            "IRD1.REG/<.AND[<NEXT/@1>,7FF]>,CNT0.REG/<.AND[<NEXT/@2>,7FF]>,CNT1.REG/<.AND[<NEXT/@3>,7FF]>,
;5276                             VREG/<.EQL[ 0, <.AND[<NEXT/@1>,3800]> , <.AND[<NEXT/@2>,3800]> , <.AND[<NEXT/@3>,3800]> ]>"
;5277
;5278    MEM [] [] []            "IRD1.MEM/<.AND[<NEXT/@1>,7FF]>,CNT0.MEM/<.AND[<NEXT/@2>,7FF]>,CNT1.MEM/<.AND[<NEXT/@3>,7FF]>,
;5279                             VMEM/<.EQL[ 0, <.AND[<NEXT/@1>,3800]> , <.AND[<NEXT/@2>,3800]> , <.AND[<NEXT/@3>,3800]> ]>"
;5280
;5281    .DCODE
;5282    SIZF [] [] [] [] [] []  "OS1/@1,OS2/@2,OS3/@3,OS4/@4,OS5/@5,OS6/@6"
;5283
;5284    .CREF
;5285    .CREF                   ;END OF NEVER CREF THE FOLLOWING WILL ALWAYS CREF
;5286    .UCODE
                                ;5287   .BIN
```

```
                                        ;5288  .TOC "INIT.MIC"
                                        ;5289  .TOC "REVISION 25.0"
                                        ;5290  ;       C. McDowell, Brian Allison, B. WARD
                                        ;5291
;5292   .NOBIN
;5293   .TOC "  Revision History"
;5294   ; REV   EXPLANATION
;5295   ; 25    Fix the 2 changes made in Rev 24
;5296   ;       REassign locations to get return -19 working
;5297   ;       Change BO.INIT_UBA_MAPS to be a subroutine that is called on
;5298   ;       both cold and warm boots.
;5299   ; 24    Change boot code to init the last 16 unibus addresses as not valid.
;5300   ;       Change boot code to make B/number work without specifying the device.
;5301   ; 23    29-MAY-1980
;5302   ;       Fix INIT code to invalidate 1st two locations in the TB.
;5303   ; 22    Turn off the cache when doing memory test for booting.
;5304   ;       Initialize the UNIBUS in the INIT code.
;5305   ;       Clear TU58 control and status registers in INIT.
;5306   ;       Add BUS/PRB.RD.PTE to avoid machine hang clearing TB and CACHE.
;5307   ;  21   Initial release.
                                        ;5308          .BIN
```

```
                                ;5309  .TOC     ""      Power Up                                : Power Up''
                                ;5310
                                ;5311  .REGION/INIT.R1L,INIT.R1H/INIT.R2L,INIT.R2H/INIT.R3L,INIT.R3H
                                ;5312  .CHANGE/INIT=1
                                ;5313  ;*************************************************************************
                                ;5314  ;       The hardware forces control to micro location 0 on power-up.
                                ;5315  ;       The microcode waits 70ms for machine stabilization and then
                                ;5316  ;       procedes when ACLO is deasserted.
                                ;5317  ;       The microcode then tests the front panel switches to determine
                                ;5318  ;       how to start up VMS.
                                ;5319  ;*************************************************************************
                                ;5320  0:
                                ;5321  BO.POWER_UP:
                                ;5322          ;-------------------------------;
                                ;5323          IO RESET,                       ; DO IO RESET FOR 70MS
U 0000, 0480,0036,4030,0043,0080,F ;5324       NOP                             ; FOR RDM
                                ;5325
                                ;5326          ;-------------------------------;
                                ;5327          M[TEMP0]_ZLIT16[8],             ; GET COUNTER FOR 70MS WAIT
U 080F, 0D86,0D37,0030,4043,0080,0 ;5328       IO RESET                        ; DO IO RESET FOR 70MS
                                ;5329
                                ;5330  =0
                                ;5331  BO.70MS_WAIT:
                                ;5332          ;0-------------------------------;
                                ;5333          M[TEMP0]_MB-ZLIT0[1],           ; DEC COUNTER
                                ;5334          IO RESET,                       ; DO IO RESET FOR 70MS
U 0800, 0186,0C10,0A30,0843,0880,0 384* ;5335  WX.EQ.0?,NEXT/BO.70MS_WAIT       ;
                                ;5336
                                ;5337  BO.TEST_ACLO:
                                ;5338          ;1-------------------------------;
                                ;5339          ACLO FPLOCK?                    ; CHECK ACLO
U 0801, 0080,0036,4CB0,0047,0081,9 ;5340  =000
                                ;5341  =001    ;001-------------------------------; ACLO OK
                                ;5342          CLEAR FLAG0,                    ; ARGUMENT FOR SUBROUTINE
U 0819, 0400,0036,4030,0047,0481,F ;5343       PUSH,NEXT/BO.COLD_START_FLAG    ; GO CLEAR COLD START FLAG
                                ;5344
                                ;5345  =011    ;011-------------------------------;
U 081B, 0480,0036,4030,0047,0080,1 ;5346       NEXT/BO.TEST_ACLO               ; WAIT FOR AC TO STABALIZE
                                ;5347
                                ;5348  =100    ;100-------------------------------;
U 081C, 0C80,0036,4030,0047,0562,C ;5349       PUSH,NEXT/MV.TEST               ; DO MICRO VERIFY
                                ;5350
                                ;5351          ;101-------------------------------;
                                ;5352          R[R5]_0,                        ; CLEAR BOOT CONTROL FLAGS
                                ;5353          CLEAR FLAG2,                    ; TELL INIT TO INIT SCBB
U 081D, 0814,05B7,0035,4047,0487,9 ;5354       PUSH,NEXT/IN.INIT               ; DO INIT
                                ;5355
                                ;5356          ;110-------------------------------;
                                ;5357          PC_ZLIT0[2],                    ; SO THE CONSOLE WILL PRINT 0 ON HALT
U 081F, 0580,0C37,0D30,1487,0080,4 ;5358       FPS1?,NEXT/BO.ACTION_SWITCH     ; CHECK BOOT ACTION SWITCH
                                ;5359  =
```

```
                                ;5360  =0000
                                ;5361  =0010    ;0010----------------------------; CLEAR BOOT CONTROL FLAGS
U 0802, 0884,05B7,0035,4047,0080,6      ;5362           R[R5]_0,NEXT/BO.BOOT            ; DO A COLD START
                                ;5363
                                ;5364           ;0011----------------------------;
                                ;5365           M[FPDOFFSET]_ZLIT0[11],         ; SET HALT CODE
U 0803, 0586,CC37,0030,8847,0000,1      ;5366           NEXT/CN.CONSOLE                 ; AND GO TO CONSOLE
                                ;5367
                                ;5368  BO.ACTION_SWITCH:
                                ;5369           ;0100----------------------------; RESTART/BOOT
                                ;5370           R[R12]_M[FPDOFFSET],            ; LOAD HALT CODE INTO AP
U 0804, 0084,C592,4037,0047,0487,4      ;5371           PUSH,NEXT/BO.FIND_RPB_SUB       ; GO GET RPB
                                ;5372
                                ;5373  BO.RESTART_HALT:
                                ;5374           ;0101----------------------------; RESTART/HALT
                                ;5375           R[R12]_M[FPDOFFSET],            ; LOAD HALT CODE INTO AP
U 0805, 0084,C592,4037,0047,0487,4      ;5376           PUSH,NEXT/BO.FIND_RPB_SUB       ; GO GET RPB
                                ;5377
                                ;5378  BO.BOOT:
                                ;5379           ;0110----------------------------; BOOT
                                ;5380           WB_RNUM,                        ; TEST FLAG FOR /X COMMAND
U 0806, 0881,C036,42B0,0047,0080,C      ;5381           WB<0>?,NEXT/BO.BOOT1            ; (RUNM BIT 0) 0=NO 1=YES
                                ;5382
                                ;5383           ;0111----------------------------; HALT
                                ;5384           M[FPDOFFSET]_ZLIT0[16],         ; SET HALT CODE
U 0807, 0D86,CC37,0030,B047,0000,1      ;5385           NEXT/CN.CONSOLE                 ; AND LET THE USER DECIDE WHAT TO DO
                                ;5386
                                ;5387           ;1000----------------------------;
                                ;5388           WB_M[MDR],                      ; CHECK WARM START BIT
U 0808, 0881,2592,42B0,0047,0082,2      ;5389           WB<0>?,NEXT/BO.R-B_WARM_CHECK   ;
                                ;5390
                                ;5391           ;1001----------------------------;
                                ;5392           WB_M[MDR],                      ; CHECK WARM START BIT
U 0809, 0881,2592,42B0,0047,0080,A      ;5393           WB<0>?,NEXT/BO.R-H_WARM_CHECK   ;
                                ;5394  =
                                ;5395  =0
                                ;5396  BO.R-H_WARM_CHECK:
                                ;5397           ;0-------------------------------;
                                ;5398           M[TEMP5]_D+ZLIT8[2],            ; GET STARTING SP
U 080A, 0186,5DB1,0030,1047,00BB,D      ;5399           NEXT/BO.RESTART                 ;
                                ;5400
                                ;5401           ;1-------------------------------;
                                ;5402           M[FPDOFFSET]_ZLIT0[12],         ; SET HALT CODE
U 080B, 0586,CC37,0030,9047,0000,1      ;5403           NEXT/CN.CONSOLE                 ; AND GO TO CONSOLE
                                ;5404
                                ;5405  =0
                                ;5406  BO.R-B_WARM_CHECK:
                                ;5407           ;0-------------------------------;
                                ;5408           M[TEMP5]_D+ZLIT8[2],            ; GET STARTING SP
U 0822, 0186,5DB1,0030,1047,00BB,D      ;5409           NEXT/BO.RESTART                 ;
                                ;5410
                                ;5411  BO.BOOT.CLR.FLAGS:
                                ;5412           ;1----------------------------; CLEAR BOOT CONTROL FLAGS
U 0823, 0884,05B7,0035,4047,0080,6      ;5413           R[R5]_0,NEXT/BO.BOOT            ; DO A COLD START
```

```
                                ;5414  0BBD:
                                ;5415  BO.RESTART:
                                ;5416          ;*******FORCE ADDRESS**********;
                                ;5417          R[SP]_M[TEMP5],                 ; LOAD STARTING SP
U 0BBD, 0484,5592,4037,8047,0483,9 ;5418       PUSH,NEXT/BO.INIT_UBA_MAPS      ; INIT UNIBUS MAPS
                                ;5419
                                ;5420  0BA0:   ;*******FORCE ADDRESS**********;
                                ;5421          PC_M[TEMP2],                    ; POINT PC TO RESTART ROUTINE
U 0BA0, 0C80,2002,403D,8487,0083,1 ;5422       NEXT/BO.IRD1                    ; GO DO THE IRD1
                                ;5423
                                ;5424  =00
                                ;5425  BO.BOOT1:
                                ;5426          ;00-----------------------------; DO MICRO VERIFY
U 080C, 0C80,0036,4030,0047,0562,C ;5427       PUSH,NEXT/MV.TEST               ;
                                ;5428
                                ;5429          ;01-----------------------------; DO INIT
                                ;5430          SET FLAG2,                      ; TELL INIT TO NOT TOUCH SCBB
                                ;5431          R[R3]_0,                        ; BOOT UNIT NUMBER PARAMATER
U 080D, 0054,05B7,0034,C047,0487,9 ;5432       PUSH,NEXT/IN.INIT               ;
                                ;5433
                                ;5434          ;10-----------------------------;
U 080E, 0780,07F8,6BFF,F847,0081,1 ;5435       LONLIT_[0F28000]                ; LOAD DEFAULT MBA ADDRESS
                                ;5436
                                ;5437  =       ;-------------------------------;
U 0811, 0C86,05BE,403D,4047,0082,E ;5438       M[TEMP0]_R[LONLIT]              ; MOVE LONLIT TO A USEFUL PLACE
                                ;5439
                                ;5440  =0      ;0-----------------------------;
                                ;5441          R[R1]_M[TEMP0],                 ; LOAD MBA ADDRESS
U 082E, 0484,0592,4034,4047,0482,B ;5442       PUSH,NEXT/BO.BOOT_SUB           ; CALL BOOTSUB
                                ;5443
                                ;5444          ;1-----------------------------;
U 082F, 0B80,07F8,00FF,F847,0081,3 ;5445       LONLIT_[0FFE000]                ; ADDRESS OF UNIBUS I/O PAGE
                                ;5446
                                ;5447          ;-------------------------------;
U 0813, 0C86,05BE,403D,4047,0081,8 ;5448       M[TEMP0]_R[LONLIT]              ; PUT LONLIT IN A USEFUL PLACE
                                ;5449
                                ;5450          ;-------------------------------;
                                ;5451          R[R2]_M[TEMP0],                 ; LOAD ADDRESS OF UNIBUS I/O PAGE
U 0818, 0C84,0592,4CF4,8047,0081,4 ;5452       FPS2?                           ; BUT ON BOOT DEVICE SWITCH
```

```
                                        ;5453  =00     ;00------------------------------; D
                                        ;5454          PC_M[PC]+ZLIT4[30],             ; POINT PC TO RIGHT ROM
U 0814, 0981,ADD1,0031,8487,0081,7      ;5455          NEXT/BO.CHECK_ROM               ;
                                        ;5456
                                        ;5457          ;01------------------------------; C
                                        ;5458          PC_M[PC]+ZLIT4[20],             ; POINT PC TO RIGHT ROM
U 0815, 0D81,ADD1,0031,0487,0081,7      ;5459          NEXT/BO.CHECK_ROM               ;
                                        ;5460
                                        ;5461          ;10------------------------------; B
                                        ;5462          PC_M[PC]+ZLIT4[10],             ; POINT PC TO RIGHT ROM
U 0816, 0D81,ADD1,0030,8487,0081,7      ;5463          NEXT/BO.CHECK_ROM               ;
                                        ;5464
                                        ;5465  BO.CHECK_ROM:
                                        ;5466          ;11------------------------------; A
                                        ;5467          WB_SEXT(XB).XOR.MINUS1 PC_PC+2, ; CHECK FOR NON-EXISTENT ROM
U 0817, 0081,7E5F,4A50,2047,0083,0      ;5468          WX.NE.0?                        ;
                                        ;5469
                                        ;5470  =00     ;00------------------------------;
                                        ;5471          M[FPDOFFSET]_ZLIT0[14],         ; PASS 'NO ROM' ERROR CODE
U 0830, 0986,CC37,0030,A047,0000,1      ;5472          NEXT/CN.CONSOLE                 ;
                                        ;5473
                                        ;5474  BO.IRD1:
                                        ;5475          ;01------------------------------;
                                        ;5476          M[FPDOFFSET]_ZLIT0[0],          ; DON'T CONFUSE CHARLIE WITH FPDOFFSET
U 0831, 0586,CC37,0030,0047,0481,A      ;5477          PUSH,NEXT/BO.IRD1_SUB           ; DO SUB CALL FOR RTUT RETURN
                                        ;5478
                                        ;5479          ;10------------------------------;
U 0832, 0C80,0036,4030,0047,0072,2      ;5480          NEXT/CO.NOP                     ; WASTE A CYCLE AND THEN IRD1
                                        ;5481  =
                                        ;5482  BO.IRD1_SUB:
                                        ;5483          ;------------------------------;
                                        ;5484          RETURN AND INHIBIT DESTINATIONS,; TELL THE HARDWARE ABOUT POWER UP
U 081A, 0C80,0036,40F0,0047,0000,1      ;5485          NEXT/1                          ; FORCE A RETURN +1
```

```
                                    ;5486  BO.COLD_START_FLAG:
                                    ;5487          ;--------------------------------;
U 081F, 0980,0D37,0036,0267,0082,5  ;5488          TRAR_ZLIT16[0C0]                 ; ADDRESS TU58 CSR
                                    ;5489
                                    ;5490          ;--------------------------------;
                                    ;5491          M[TEMP6]_TU58REGS,               ; GET CURRENT CONTENTS OF CSR
U 0825, 0086,6036,4430,03E7,0172,8  ;5492          FLAG0?                           ; TEST FOR SET OR CLEAR COLD BIT
                                    ;5493
                                    ;5494  1728:    ;0******FORCE ADDRESS**********; CLEAR COLD START BIT
                                    ;5495          TU58REGS_M[TEMP6].AND.0LIT16[17F],;
U 1728, 0980,6F12,00BB,FAE7,0000,3  ;5496          RETURN [3]                       ;
                                    ;5497
                                    ;5498  1729:    ;1******FORCE ADDRESS**********;
                                    ;5499          WB_M[TEMP6].AND.ZLIT16[80],      ; TEST COLD START BIT
U 1729, 0180,6D12,0A34,0047,0085,0  ;5500          WX.EQ.0?                         ;
                                    ;5501
                                    ;5502  =0       ;0--------------------------------;
                                    ;5503          M[FPDOFFSET]_ZLIT0[15],          ; SET HALT CODE
U 0850, 0D86,CC37,0030,A847,0000,1  ;5504          NEXT/CN.CONSOLE                  ; AND GO TO CONSOLE
                                    ;5505
                                    ;5506          ;1--------------------------------; SET COLD START BIT
                                    ;5507          TU58REGS_M[TEMP6].OR.ZLIT16[80],;
U 0851, 0180,6D12,40B4,02E7,0000,3  ;5508          RETURN [3]                       ;
```

```
                                ;5509  .TOC   "       Boot                                    : Boot Sub"
                                ;5510
                                ;5511  ;*****************************************************************************
                                ;5512  ;*****************************************************************************
                                ;5513  BO.BOOT_SUB:
                                ;5514          ;--------------------------------;
U 082B, 0186,CC77,0031,4047,0072,1  ;5515          M[FPDOFFSET]_ZLIT28[28]          ; CLAIM DISPTACH SLOT 2 ON MACH CHECK
                                ;5516
                                ;5517  721:    ;*********FORCE ADDRESS*********;
                                ;5518          ;--------------------------------;
U 0721, 0D80,0CB7,0030,3687,007D,D  ;5519          MEMSCAR_ZLIT24[6]                ; LOAD ADDRESS OF CACHE DISABLE REGISTER
                                ;5520
                                ;5521  7DD:    ;*********FORCE ADDRESS*********;
                                ;5522          ;--------------------------------;
U 07DD, 0D80,0CB7,0030,0E07,0083,3  ;5523          MEMSCR_ZLIT24[1]                 ; DISABLE THE CACHE FOR MEMORY TEST.
                                ;5524
                                ;5525          ;--------------------------------;
U 0833, 0086,05BE,603D,84A7,0085,2  ;5526          VA_G_M[TEMP0]_R[ZERO]            ; STARTING ADDRESS FOR READS
                                ;5527  =0
                                ;5528  BO.START_SEARCH:
                                ;5529          ;0-------------------------------;
                                ;5530          M[TEMP1]_ZLIT8[40],              ; NUMBER OF WORDS TO READ (64K)
U 0852, 0D86,1DB7,0032,0047,0083,4  ;5531          NEXT/BO.INITIAL_READ             ;
                                ;5532
                                ;5533          ;1-------------------------------;
                                ;5534          M[FPDOFFSET]_ZLIT0[13],          ; SET UP HALT CODE
U 0853, 0186,CC37,0030,9847,0000,1  ;5535          NEXT/CN.CONSOLE                  ;
                                ;5536
                                ;5537  BO.DEC_WORD_COUNT:
                                ;5538          ;--------------------------------;
                                ;5539          M[TEMP1]_MB-ZLIT0[1],            ; DEC "WORDS LEFT TO READ" COUNT
U 0837, 0586,1C10,0A30,0847,0883,4 384*  ;5540          WX.EQ.0?                         ; 64K FOUND???
                                ;5541
                                ;5542  =00
                                ;5543  BO.INITIAL_READ:
                                ;5544          ;00------------------------------;
                                ;5545          READ,SIZE[LONG],                 ; ACCESS A LONGWORD
                                ;5546          VA_R[TEMP0]_RB+CONX(4),          ; POINT ADDRESS TO NEXT WORD
U 0834, 0084,073D,0020,04B0,0083,7  ;5547          NEXT/BO.DEC_WORD_COUNT           ; IF THIS READ FAILS CONTROL WILL BE
                                ;5548                                           ; PASSED THROUGH SLOT 2 OF MACH CHECK
                                ;5549                                           ; DISPATCH TABLE
                                ;5550
                                ;5551          ;01------------------------------;
                                ;5552          VA_R[TEMP0]_RB-CONX(4),          ; POINT ADDRESS TO HIGHEST WORD READ
U 0835, 0485,273C,0020,04A7,0082,C  ;5553          MB_M[MDR],NEXT/BO.WRITE_ZERO     ; GO START TESTING HIGHEST WORD
                                ;5554
                                ;5555          ;10------------------------------;
U 0836, 0480,0036,4030,0047,0084,6  ;5556          NEXT/BO.RESTART_SEARCH1          ; CATCH MACHINE CHECK RETURN
                                ;5557  =
```

```
                                ;5558  =000    ;000------------------------------;
                                ;5559          Q_ZLIT0[1],                     ; GET INITIAL 1 TO WALK
U 0828, 0580,0C37,1030,0847,0083,A ;5560       NEXT/B0.WRITE_WALK1             ;
                                ;5561
                                ;5562  =010    ;010------------------------------;
                                ;5563          Q_-1.WRITE,SIZE[LONG],          ; WRITE -1 AND LEAVE IN Q FOR READ SUB
U 082A, 0480,0E77,1020,05D8,0487,1 ;5564       PUSH,NEXT/B0.READ_SUB           ; CALL READ SUB  (RETURN -2)
                                ;5565  =100
                                ;5566  B0.WRITE_ZERO:
                                ;5567          ;100-----------------------------;
                                ;5568          Q_R[ZERO],WRITE,SIZE[LONG],     ; WRITE 0 AND LEAVE IN Q FOR READ SUB
U 082C, 0480,05BE,502D,85D8,0487,1 ;5569       PUSH,NEXT/B0.READ_SUB           ; CALL READ SUB  (RETURN -2)
                                ;5570  =
                                ;5571  =00
                                ;5572          ;00------------------------------;
                                ;5573          Q_(R[ZERO]+Q).SL.1,             ; WALK THE ONE
U 0838, 0480,0039,DA3D,8047,0883,A 362* ;5574  WX.EQ.0?                        ; LOOK FOR 1 WALKING OFF HIGH END
                                ;5575  =10
                                ;5576  B0.WRITE_WALK1:
                                ;5577          ;10------------------------------;
                                ;5578          WRITE Q,SIZE[LONG],             ; WRITE THE WALKING 1
U 083A, 0080,003A,402D,85D8,0487,1 ;5579       PUSH,NEXT/B0.READ_SUB           ; CHECK THE WRITE (RETURN -2)
                                ;5580
                                ;5581          ;11------------------------------;
                                ;5582          Q_OLIT0[1FE],                   ; GET STARTING 0 TO WALK  Q<-FFFFFFFE
U 083B, 0980,0E37,103F,F047,0083,F ;5583       NEXT/B0.WRITE_WALK0             ; GO WALK 0
                                ;5584  =00
                                ;5585  =01     ;01------------------------------;
                                ;5586          Q_(R[ZERO]+Q).RR.1,             ; WALK THE ZERO ONE NOTCH
U 083D, 0880,0339,92BD,8047,0883,E 355* ;5587  WB<0>?                          ; TEST FOR COMPLETE WALK
                                ;5588
                                ;5589          ;10------------------------------;
                                ;5590          M[TEMP5]_D+ZLIT8[2],PUSH,       ; POINT SP TO BOTTOM OF 2ND GOOD PAGE
U 083E, 0986,5DB1,0030,1047,0483,9 ;5591       NEXT/B0.INIT_UBA_MAPS           ; INIT UNIBUS MAPS (RET -19)
                                ;5592
                                ;5593  B0.WRITE_WALK0:
                                ;5594          ;11------------------------------;
                                ;5595          WRITE Q,SIZE[LONG],             ; WRITE THE WALKING 0
U 083F, 0080,003A,402D,85D8,0487,1 ;5596       PUSH,NEXT/B0.READ_SUB           ; CHECK THE WRITE (RETURN -2)
                                ;5597
                                ;5598  B0.INIT_UBA_MAPS:
                                ;5599          ;------------------------------;
                                ;5600          M[FPDOFFSET]_ZLIT28[48],        ; CLAIM DISPATCH SLOT 4 ON MACH CHECK
U 0839, 0906,CC77,0032,4047,007D,E ;5601       CLEAR FLAG0                     ; FLAG TO CONTROL LOOP EXECUTION
                                ;5602
                                ;5603  7DE:    ;*********FORCE ADDRESS*********;
                                ;5604          ;------------------------------;
U 07DF, 0980,0CB7,0030,0607,0083,C ;5605       MEMSCR_ZLIT24[0]                ; RE-ENABLE THE CACHE AFTER MEMORY TEST.
                                ;5606
                                ;5607          ;------------------------------;
U 083C, 0880,07F8,67BF,F847,0084,3 ;5608       LONLIT_[0F30800]                ; ADDRESS OF FIRST UBA
```

```
                                        ;5717   .TOC     ""      Boot                                    : Find RPB Sub''
                                        ;5718
                                        ;5719   ;*****************************************************************************
                                        ;5720   ;*****************************************************************************
                                        ;5721   BO.FIND_RPB_SUB:
                                        ;5722           ;-------------------------------;
U 0874, 0D86,CC77,0032,C047,0087,5      ;5723           M[FPDOFFSET]_ZLIT28[58]         ; CLAIM DISPATCH SLOT 5 ON MACH CHECK
                                        ;5724
                                        ;5725           ;-------------------------------;
U 0875, 0C80,05BE,703D,84A7,0085,6      ;5726           VA_Q_D_R[ZERO]                  ; STARTING ADDRESS FOR RPB SEARCH
                                        ;5727   =0
                                        ;5728   BO.READ_RPB_HEADER:
                                        ;5729           ;0-------------------------------;
                                        ;5730           READ,SIZE[LONG],                ; GET FIRST WORD OF RPB (MAYBE)
                                        ;5731           M[TEMP1]_ZLIT0[1F],             ; GET CHECKSUM COUNTER
                                        ;5732           VA_VA+4,                        ; IF THIS READ FAILS CONTROL WILL BE
U 0856, 0D86,1C37,0020,FC50,0085,8      ;5733           NEXT/BO.CHECK_RPB               ; PASSED THROUGH SLOT 5 OF MACH CHECK
                                        ;5734                                           ; DISPATCH TABLE
                                        ;5735
                                        ;5736           ;1-------------------------------;
U 0857, 0880,0036,40B0,0047,03FF,E      ;5737           RETURN [-2]                     ; RETURN WITH FAILURE
                                        ;5738   =0
                                        ;5739   BO.CHECK_RPB:
                                        ;5740           ;0-------------------------------;
                                        ;5741           WB_M[MDR]-Q,                    ; CHECK FOR START OF RPB
U 0858, 0481,2008,0A30,0047,0885,A 344* ;5742           WX.EQ.0?,NEXT/BO.BAD_RPB        ;
                                        ;5743
                                        ;5744           ;1-------------------------------;
U 0859, 0C80,0036,4030,0047,0087,8      ;5745           NEXT/BO.BAD_RPB1                ; CATCH THE MACHINE CHECK
                                        ;5746   =0
                                        ;5747   BO.BAD_RPB:
                                        ;5748           ;0-------------------------------; BAD RPB (TRY AGAIN)
                                        ;5749           VA_Q_D_D+ZLIT8[2],              ; POINT TO NEXT PAGE
U 085A, 0180,0DB1,3030,14A7,0087,8      ;5750           NEXT/BO.BAD_RPB1                ;
                                        ;5751
                                        ;5752           ;1-------------------------------;
                                        ;5753           READ,SIZE[LONG],                ; GET STARTING ADD OF RESTART ROUTINE
U 085B, 0886,05BE,402D,8050,0085,C      ;5754           M[TEMP0]_R[ZERO]                ; INIT RPB CHECKSUM ACC
                                        ;5755                                           ; IF THIS READ FAILS CONTROL WILL BE
                                        ;5756                                           ; PASSED THROUGH SLOT 5 OF MACH CHECK
                                        ;5757                                           ; DISPATCH TABLE
                                        ;5758   =0
                                        ;5759           ;0-------------------------------;
                                        ;5760           VA_R[TEMP2]_M[MDR],             ; SAVE STARTING ADD OF RESTART ROUTINE
                                        ;5761           WX.NE.0?,                       ; MAKE SURE IT'S NOT 0
U 085C, 0485,2592,4A70,84A7,0085,E      ;5762           NEXT/BO.CHECK_RESTART_ADDRESS   ;
                                        ;5763
                                        ;5764           ;1-------------------------------;
U 085D, 0C80,0036,4030,0047,0087,8      ;5765           NEXT/BO.BAD_RPB1                ; CATCH THE MACHINE CHECK
```

```
                                                ;5766  =0
                                                ;5767  BO.CHECK_RESTART_ADDRESS:
                                                ;5768          ;0--------------------------------;
 085E, 0080,0036,4030,0047,0085,A               ;5769          NEXT/BO.BAD_RPB                   ; TRY AGAIN
                                                ;5770
                                                ;5771          ;1--------------------------------;
 085F, 0480,0036,4030,0047,0086,2               ;5772          NEXT/BO.READ_RESTART_ROUTINE      ; ADDRESS OK
                                                ;5773  =0
                                                ;5774  BO.CSUM_RESTART_ROUTINE:
                                                ;5775          ;0--------------------------------;
                                                ;5776          R[TEMP0]_M[MDR]+RB,               ; DO CHECKSUM
 0860, 0085,2001,0030,0047,0087,6               ;5777          NEXT/BO.DEC_CSUM_COUNTER          ;
                                                ;5778
                                                ;5779          ;1--------------------------------;
 0861, 0080,0036,4030,0047,0087,8               ;5780          NEXT/BO.BAD_RPB1                  ; CATCH THE MACHINE CHECK
                                                ;5781
                                                ;5782  BO.DEC_CSUM_COUNTER:
                                                ;5783          ;--------------------------------;
                                                ;5784          M[TEMP1]_MB-ZLIT0[1],             ; DEC NUMBER OF WORDS TO CHECKSUM
 0876, 0586,1010,0A30,0847,0886,2 384*          ;5785          WX.EQ.0?                          ; ALL DONE
                                                ;5786  =0
                                                ;5787  BO.READ_RESTART_ROUTINE:
                                                ;5788          ;0--------------------------------;
                                                ;5789          READ,SIZE[LONG],                  ; START READING RESTART ROUTINE
                                                ;5790          VA_VA+4,                          ; POINT TO NEXT WORD
 0862, 0080,0036,4020,0450,0086,0               ;5791          NEXT/BO.CSUM_RESTART_ROUTINE      ; IF THIS READ FAILS CONTROL WILL BE
                                                ;5792                                            ; PASSED THROUGH SLOT 5 OF MACH CHECK
                                                ;5793                                            ; DISPATCH TABLE
                                                ;5794
                                                ;5795          ;1--------------------------------; ALL DONE READING RESTART ROUTINE
 0863, 0180,0031,0030,44A7,0087,7               ;5796          VA_D+ZLIT0[8]                     ; GET ADDRESS OF CHECKSUM WORD
                                                ;5797
                                                ;5798          ;--------------------------------;
                                                ;5799          READ,SIZE[LONG],                  ; READ CHECKSUM WORD
 0877, 0880,0036,4020,0450,0086,4               ;5800          VA_VA+4                           ; POINT VA TO WARM START BIT
                                                ;5801                                            ; IF THIS READ FAILS CONTROL WILL BE
                                                ;5802                                            ; PASSED THROUGH SLOT 5 OF MACH CHECK
                                                ;5803                                            ; DISPATCH TABLE
                                                ;5804
                                                ;5805  =0      ;0--------------------------------;
                                                ;5806          WB_M[MDR]-R[TEMP0],               ; CHECK CHECKSUM
 0864, 0081,2000,0A30,0047,0886,6 345*          ;5807          WX.EQ.0?,NEXT/BO.CHECK_CHECKSUM   ;
                                                ;5808
                                                ;5809          ;1--------------------------------;
 0865, 0C80,0036,4030,0047,0087,8               ;5810          NEXT/BO.BAD_RPB1                  ; CATCH THE MACHINE CHECK
                                                ;5811  =0
                                                ;5812  BO.CHECK_CHECKSUM:
                                                ;5813          ;0--------------------------------;
 0866, 0C80,0036,4030,0047,0085,A               ;5814          NEXT/BO.BAD_RPB                   ; CONTINUE LOOKING
                                                ;5815
                                                ;5816          ;1--------------------------------;
                                                ;5817          READ,SIZE[LONG],                  ; GET WARM START BIT
 0867, 0C80,0036,40A0,0050,0000,4               ;5818          RETURN [+4]                       ; IF THIS READ FAILS CONTROL WILL BE
                                                ;5819                                            ; PASSED THROUGH SLOT 5 OF MACH CHECK
                                                ;5820                                            ; DISPATCH TABLE
```

```
                                        ;5821  BO.BAD_RPB1:
                                        ;5822          ;--------------------------------;
                                        ;5823          WB_M[VA].AND.ZLIT16[80],         ; SEE IF BEYOND PHYSICAL MEMORY YET
U 0878, 0581,BD12,0A74,0047,0085,6      ;5824          WX.NE.0?,NEXT/BO.READ_RPB_HEADER;
                                        ;5825
                                        ;5826  .CHANGE/INIT=0
                                        ;5827  ; ************ THIS WORD GOES INTO MACH CHECK DISPATCH TABLE SLOT #5
                                        ;5828  ;       ;--------------------------------;
                                        ;5829  ;       VA_Q_D_D+ZLIT8[2],               ; POINT TO NEXT PAGE
                                        ;5830  ;       RETURN[+1]                       ;
                                        ;5831  ; ************
```

```
;5832  .TOC '  Initialize Microcode for the Console and Power up''
;5833
;5834  ;*************************************************************************
;5835  ;       INIT.MIC        INITIALIZATION IS CALLED BY THE CONSOLE AND AT POWER UP.
;5836  ;               RESOURCES               LONLIT
;5837  ;                                       FLAG2 CLEAR IF POWER UP
;5838  ;                                             SET IF CONSOLE
;5839  ;                                       FLAG0 WHETHER OR NOT POWER UP
;5840  ;                                       CRAR
;5841  ;                                       DREG
;5842  ;                                       VA
;5843  ;               OUTPUT                  PSL             41F0000
;5844  ;                                       IPL             1F
;5845  ;                                       ICBB            -1 (AT POWER UP ONLY)
;5846  ;                                       ASTLVL          4
;5847  ;                                       SISR            0
;5848  ;                                       FPDOFFSET       3
;5849  ;                                   ** RCSR             0
;5850  ;                                   ** XCSR<6>          0
;5851  ;                                   ** MME              0 (SET WHEN PRINIT)
;5852  ;                                      PME              0
;5853  ;                                      CACHE            INVALIDATED
;5854  ;                                      TB               INVALIDATED
;5855  ;                                   ** ICCS             0
;5856  ;                                      PC               0
;5857  ;                                      XB               FLUSHED WHEN PC_0
;5858  ;                                      SOFTIPR          0
;5859  ;                                      PROCESS INIT IS ALSO DONE
;5860  ;               SUBROUTINES            IN.CLR.CACHE.ROUT       CLEARS THE CACHE
;5861  ;                                      MP.MTPR.TBIA20  CLEARS THE TB
;5862  ;
;5863  ;
;5864  ;** I ASSUME RXCS IN THE SRM IS THE SAME AS RCSR IN DEFIN.
;5865  ;     AND      TXCS IN THE SRM IS THE SAME AS XCSR IN DEFIN.
;5866  ;     AND      MAPEN IN THE SRM IS THE SAME AS MME IN DEFIN.
;5867  ;     AND      ICCS IN THE SRM IS THE SAME AS TCSR IN DEFIN.
;5868  ;
;5869  ;        A PROCESS INIT BUS FUNCTION IS DONE.
;5870  ;        EVERYTHING ELSE MENTIONED IN THE SRM SECTION 9.7
;5871  ;        IS EITHER INITIALIZED BY THE HARDWARE OR UNPREDICTABLE.
;5872  ;*************************************************************************
```

```
                                ;5873  IN.INIT:
                                ;5874          ;--------------------------------;
                                ;5875          LONLIT_[41F0000],               ;LONLIT GETS 41F0000
U 0879, 0F80,07DF,07FF,F847,0087,C ;5876       NEXT/IN.PSL.LONLIT              ;GOTO REG FLOW
                                ;5877
                                ;5878  IN.PC_0:
                                ;5879          ;--------------------------------;
                                ;5880          PC_R[ZERO],                     ;PC GETS 0
                                ;5881          CLEAR FLAG1,                    ;FOR CHARLIE'S CLEAR TB SUBR
U 087A, 0008,05BE,40BD,8487,0000,1 ;5882       RETURN [1]                      ;RETURN+1
                                ;5883
                                ;5884  IN.VA_0:
                                ;5885          ;--------------------------------;
                                ;5886          COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG CLEARING CACHE
                                ;5887          VA_R[ZERO],                     ;VA GETS 0
U 087B, 0080,05BE,40BD,84B7,0000,1 ;5888       RETURN [1]                      ;RETURN+1
                                ;5889
                                ;5890  IN.PSL.LONLIT:
                                ;5891          ;--------------------------------;
U 087C, 0880,05BF,403D,4007,0086,8 ;5892       PSL_R[LONLIT]                   ;PSL GETS LONLIT
                                ;5893
                                ;5894  =0
                                ;5895          ;0-------------------------------;
                                ;5896          PUSH,                           ;JSR
                                ;5897          STEPC_2,                        ;
                                ;5898          CRAR_ZLIT16[80],                ;CRAR GETS 2
U 0868, 05A0,0D37,0034,0247,0487,A ;5899       NEXT7IN.PC_0                    ;NOW IF WE CONWRITE
                                ;5900                                          ;WE WILL WRITE TO RXCS
                                ;5901
                                ;5902          ;1-------------------------------;
                                ;5903          CONREGS_D_M[SISR]_R[ZERO],      ;RXCS GETS 0
U 0869, 089E,F5BE,603D,82C7,00BE,E ;5904       DEC STEPC                       ;SISR GETS 0
                                ;5905
                                ;5906  0BEE:   ;*********FORCE ADDRESS*********;
U 0BEE, 0C80,05BE,403D,84A7,0084,8 ;5907       VA_R[ZERO]                      ;ZERO VA FOR CLEARING TB
                                ;5908
                                ;5909  =00     ;00------------------------------;
                                ;5910          PUSH,DEC STEPC,                 ;
                                ;5911          COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG CLEARING TB
                                ;5912          CRAR_ZLIT16[40],                ;CRAR GETS 1
U 0848, 0598,0D37,0032,0257,04E4,C ;5913       NEXT7MP.MTPR.TBIA20             ;NOW IF WE CONWRITE
                                ;5914                                          ;WE WILL WRITE TO TXCS
                                ;5915
                                ;5916          ;01------------------------------;
                                ;5917          PUSH,VA_D_ZLIT24[80],           ;
                                ;5918          COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG CLEARING TB
                                ;5919          CLEAR FLAG1,                    ;
U 0849, 0908,0CB7,2034,04B7,04E4,C ;5920       NEXT/MP.MTPR.TBIA20             ;NOW IF WE CONWRITE
                                ;5921
                                ;5922          ;10------------------------------;
                                ;5923          PUSH,NEXT/IN.CLR.CACHE.ROUT,    ;CLEAR THE CACHE ROUTINE
U 084A, 0480,05BE,403D,82C7,0484,D ;5924       CONREGS_R[ZERO]                 ;TXCS FPDOFFSET GET 0
                                ;5925
                                ;5926          ;11------------------------------;
U 084B, 0080,05B7,0030,0787,0087,D ;5927       SOFTIPR_0                       ;
```

```
                                ;5928           ;---------------------------------;
U 087D, 0880,05B7,0030,0167,0086,A ;5929        TCSR_0                          ;TCSR_0
                                ;5930
                                ;5931   =0      ;0--------------------------------;
                                ;5932           PUSH,                           ;JSR
                                ;5933           ASTLVL_[4],                     ;DONE WITH CACHE
U 086A, 0980,0CB7,0030,2707,0487,A ;5934        NEXT/IN.PC_0                    ;ASTLVL GETS 4
                                ;5935                                           ;CALL PC GETS 0
                                ;5936                                           ;THIS FLUSHES OUT XB
                                ;5937
                                ;5938           ;1--------------------------------;
                                ;5939           PME_0 FPDOFFSET_3,              ; SET POWER UP CODE FOR VMS RESTART
U 086B, 0586,CC37,0030,1A81,007D,F ;5940        PROCESS INIT                    ;PROCESS INIT
                                ;5941
                                ;5942   7DF:    ;*********FORCE ADDRESS*********;
U 07DF, 0980,0DB7,2030,1847,007E,0 ;5943        D_ZLIT8[3]                      ; LOAD COUNT FOR HOLDING IO RESET
                                ;5944   7E0:
                                ;5945   IN.IORESET:
                                ;5946           ;*********FORCE ADDRESS*********;
                                ;5947           IO RESET,D_D-1,WX.EQ.0?,        ; INITIALIZE THE UNIBUS
U 07E0, 0080,0E71,2A30,0043,087E,0 378* ;5948   NEXT/IN.IORESET                  ; FOR 200 MICRO SECONDS.
                                ;5949
                                ;5950   7E1:    ;*********FORCE ADDRESS*********;
U 07E1, 0580,0D37,0034,0267,007E,2 ;5951        TRAR_ZLIT16[80]                 ; LOAD ADDRESS OF RECEIVER DONE AND IE
                                ;5952
                                ;5953   7E2:    ;*********FORCE ADDRESS*********;
U 07E2, 0180,0D37,0030,02E7,007E,3 ;5954        TU58REGS_ZLIT16[0]              ; CLEAR DONE AND IE
                                ;5955
                                ;5956   7E3:    ;*********FORCE ADDRESS*********;
U 07E3, 0580,0D37,0032,0267,007E,4 ;5957        TRAR_ZLIT16[40]                 ; LOAD ADDRESS OF TRANS READY AND IE
                                ;5958
                                ;5959   7E4:    ;*********FORCE ADDRESS*********;
U 07E4, 0084,05B7,0033,42E7,0087,E ;5960        TU58REGS_R[TEMP13]_0            ; CLEAR READY, IE AND CONSOLE SAVED MME
                                ;5961
                                ;5962           ;---------------------------------;
U 087E, 0180,0D37,0036,0247,0087,F ;5963        CRAR_ZLIT16[0C0]                ; ADDRESS TXCSR
                                ;5964
                                ;5965           ;---------------------------------;
                                ;5966           CONREGS_ZLIT16[40],             ; CLEAR CON HALT
                                ;5967           FLAG2?,                         ;FLAG2 SAYS NOT POWER UP OR POWER UP
U 087F, 0D80,0D37,04B2,02C7,0082,9 ;5968        NEXT/IN.FLAG2.NOT.SET           ;FLAG2 SAYS WHICH WAY USER ENTERED
                                ;5969
                                ;5970   =0**
                                ;5971   IN.FLAG2.NOT.SET:
                                ;5972           ;0**------------------------------;
                                ;5973           M[SCBB]_-1,                     ;SCBB GETS -1
U 0829, 0C86,EE77,00B0,0047,0000,1 ;5974        RETURN [1]                      ;POWER UP
                                ;5975
                                ;5976           ;1**------------------------------;
                                ;5977           M[TEMP6]_ZLIT0[0D],             ;TEMP6 GETS A <CR>
U 082D, 0986,6C37,00B0,6847,0000,1 ;5978        RETURN [1]                      ;BOOTSUB OR START OR INIT OR BOOT?
```

```
                                ;5979  .TOC "  Init                                    :       IN.CLR.CACHE.ROUT"
                                ;5980  ;*************************************************************************************
                                ;5981  ;                        IN.CLR.CACHE.ROUT
                                ;5982  ;                RESOURCES               DREG
                                ;5983  ;                                        VA
                                ;5984  ;                OUTPUT                  CLEARED CACHE
                                ;5985  ;*************************************************************************************
                                ;5986  =00
                                ;5987  =01
                                ;5988  IN.CLR.CACHE.ROUT:
                                ;5989          ;01-------------------------------;
                                ;5990          PUSH,                           ;
                                ;5991          D_ZLIT8[4],                     ;TB AT END,CLEAR CACHE
U 084D, 0180,0DB7,2030,2047,0487,B ;5992          NEXT/IN.VA_0                    ;CALL VA GETS 0 RET+1
                                ;5993                                          ;D GETS 1024
                                ;5994
                                ;5995  IN.CLR.CACHE:
                                ;5996          ;10-------------------------------;
                                ;5997          VA_VA+4 CLEAR CACHE,            ;CLEAR CACHE
U 084E, 0181,BC11,0030,25A7,0088,2 ;5998          NEXT/IN.DEC.D                   ;VA GETS VA+4
                                ;5999
                                ;6000          ;11-------------------------------;
U 084F, 0880,0036,40B0,0047,0000,1 ;6001          RETURN [1]                      ;RETURN+1
                                ;6002
                                ;6003  IN.DEC.D:
                                ;6004          ;-------------------------------;
                                ;6005          COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG CLEARING CACHE
                                ;6006          D_D-ZLIT0[1],                   ;D GETS D-1
U 0882, 0180,0C30,2A30,0857,0884,E 384* ;6007          WX.EQ.0?,NEXT/IN.CLR.CACHE      ;
```

6007; This page intentionally left blank.

```
                                        ;6008  .TOC "CONSOL.MIC"
                                        ;6009  .TOC "REVISION 33.0"
                                        ;6010  ;       CHARLIE MCDOWELL
                                        ;6011
;6012   .NOBIN
;6013
;6014   .TOC "  Revision History"
;6015
;6016   ; REV   EXPLAINATION
;6017
;6018   ; 33    Fix B/number to work without specifying the device.
;6019   ; 32    22-May-1980
;6020   ;       Cleanup and add comments.
;6021   ;       Fix D P<CR> to not deposit a 0 in the PSL.
;6022   ; 31    Attempt restart for halt codes other than 6.
;6023   ;       Clear stepc before depositing to iprs (for TBIA).
;6024   ;       Fix clearing of halt pending on restart.
;6025   ;       Set MM.NOINT on command execution to avoid some interrupts getting in.
;6026   ; 30    Initial release.
                                        ;6027          .BIN
```

```
;6028  .TOC "  Console Command Parser          : Console Resource Usage"
;6029
;6030  ;*****************************************************************************
;6031  ;       Resources (unless otherwise noted these resource assignments apply to
;6032  ;                  all console microcode)
;6033  ;                       TEMP0           Counter for number buffer 0.
;6034  ;                       TEMP1           Number buffer 0 (BUF0).
;6035  ;                       TEMP2           Counter for number buffer 1.
;6036  ;                       TEMP3           Number buffer 1 (BUF1).
;6037  ;                       TEMP5           Character to be typed or last char read.
;6038  ;                       TEMP6           Command code and switch flags.
;6039  ;                       TEMP6<0>        0 if current command needs space switch
;6040  ;                       TEMP6<1>        0 if current command needs size switch
;6041  ;                       TEMP6<2>        0 if current command needs BUF0 number
;6042  ;                       TEMP6<3>        0 if current command needs BUF1 number
;6043  ;                       TEMP6<8>        0 if current command needs number switch
;6044  ;                       TEMP6<9>        0 if current command needs X switch
;6045  ;                       TEMP6<7:4>      0 - EXAMINE
;6046  ;                                       1 - DEPOSIT
;6047  ;                                       2 - HALT
;6048  ;                                       3 - NEXT
;6049  ;                                       4 - CONTINUE
;6050  ;                                       5 - START
;6051  ;                                       7 - BOOT
;6052  ;                                       8 - EXAMINE PSL
;6053  ;                                       9 - DEPOSIT PSL
;6054  ;                                       A - X BINARY TRANSFER
;6055  ;                                       B - INITIALIZE
;6056  ;                                       D - TEST
;6057  ;                       TEMP7           Checksum of characters read from console
;6058  ;                       MTEMP8          Current space switch value
;6059  ;                       RTEMP8          Saved space switch for delete
;6060  ;                       MTEMP9          Current size switch value
;6061  ;                       RTEMP9          Saved size switch for delete
;6062  ;                       RTEMP10         Default next address
;6063  ;                       RTEMP12         Save PSL
;6064  ;                       RTEMP13         Save initial MME value
;6065  ;                       RTMPGPR         Copy of PC for examine/deposit via RNUM=15.
;6066  ;                       Q               Save adjustment to RTEMP10 for '*' as address
;6067  ;                       STEPC           # of times for CN.TYPE.CHAR to type the
;6068  ;                                       character in TEMP5
;6069  ;
;6070  ;                       FLAG0           0 if current number buffer is BUF0
;6071  ;                                       1 if current number buffer is BUF1
;6072  ;                       FLAG1           0 normal
;6073  ;                                       1 signal to CN.GET.NUMBER that next
;6074  ;                                         character has already been read
;6075  ;                       FLAG2           0 normal (not APT)
;6076  ;                                       1 binary transfer mode
;6077  ;                       FLAG3           0 normal (not BOOT command)
;6078  ;                                       1 BOOT command needs device type
;6079  ;                       MM.NOINT        0 normal
;6080  ;                                       1 last character typed was <rubout>
;6081  ;*****************************************************************************
```

```
                                ;6082  .TOC "  Console Command Parser          :  Console Initialization"
                                ;6083
                                ;6084  ;*************************************************************************
                                ;6085  ;       Entry Points    16(HEX)          ^P and HALT bit set.
                                ;6086  ;                       CN.TYPE.PC       Micro code detected halts.
                                ;6087  ;
                                ;6088  ;       Input           FPDOFFSET        Low byte is used for halt code.
                                ;6089  ;                       PC               If CN.TYPE.PC entry is used PC should
                                ;6090  ;                                        contain the address of the next OPCOD
                                ;6091  ;                                        to execute plus two (ie the PC typed on
                                ;6092  ;                                        the console will be PC-2).
                                ;6093  ;                       STACK FLAG       If set front panel switches are checked
                                ;6094  ;                                        for RESTART/BOOT (cleared after check).
                                ;6095  ;
                                ;6096  ;       Output          FLAGS            Clear
                                ;6097  ;                       TEMP0-TEMP7      Zero
                                ;6098  ;                       MTEMP8,RTEMP8    Zero, default space is P
                                ;6099  ;                       MTEMP9,RTEMP9    8, default size is long (# of digits)
                                ;6100  ;                       MTEMP10,RTEMP10  Zero, default address is zero
                                ;6101  ;
                                ;6102  ;       Resources       RNUM             Used for clearing temps.
                                ;6103  ;                       TEMP7            Pass number to type to CN.TYPE.NUMBER.
                                ;6104  ;                       MTEMP10          Pass # of digits to CN.TYPE.NUMBER.
                                ;6105  ;                       (for the following see resource usage at top of console)
                                ;6106  ;                       STEPC, TEMP5, RTEMP13, RTMPGPR
                                ;6107  ;
                                ;6108  ;       Subroutines/External Labels
                                ;6109  ;                       CN.TYPE.NUMBER
                                ;6110  ;                       CN.TYPE.CHAR
                                ;6111  ;                       CN.PROMPT
                                ;6112  ;                       CN.CLEAR.TEMPS
                                ;6113  ;                       MS.HALT.MICRO
                                ;6114  ;                       BO.COLD_START_FLAG
                                ;6115  ;
                                ;6116  ;       This section types the PC and 1 byte error code (from FPDOFFSET) on
                                ;6117  ;       the console terminal, sets the HALT bit and initializes the console
                                ;6118  ;       scratch pad registers.
                                ;6119  ;*************************************************************************
                                ;6120
                                ;6121  16:     ;-------------------------------;
U 0016, 0C81,9002,403D,8487,0088,3  ;6122       PC_M[PCBACK]                    ; USE PCBACK IF FROM DO SERVICE.
                                ;6123
                                ;6124  .REGION/CONSOL.R1L,CONSOL.R1H
                                ;6125  CN.CNTRL.P.HALT:
                                ;6126          ;-------------------------------;
                                ;6127          M[FPDOFFSET]_ZLIT0[02],         ; SET HALT CODE.
U 0883, 0186,CC37,0030,1047,0000,1  ;6128       NEXT/MS.HALT.MICRO              ; GO SET HALT, WILL BRANCH TO CN.CONSOLE
```

```
                                ;6129  884:    ;*********FORCE ADDRESS*********;
                                ;6130  CN.TYPE.PC:
                                ;6131          ;--------------------------------;
U 0884, 0900,0D37,0030,0247,0088,5 ;6132       CRAR_ZLIT16[0],CLEAR FLAG0      ; LOAD ADDRESS OF CONSOLE RECEIVER BUF
                                ;6133
                                ;6134          ;--------------------------------;
U 0885, 040E,5036,4030,03C7,0089,3 ;6135       M[TEMP5]_CONREGS,CLEAR FLAG1    ; READ BUFFER TO CLEAR READY BIT
                                ;6136
                                ;6137          ;--------------------------------;
                                ;6138          PC_R[RTMPGPR]_M[PC]-CONX(2),    ; SAVE PC FOR E/D COMMANDS
U 0893, 0015,A710,0017,C487,0089,C ;6139       CLEAR FLAG2                     ;
                                ;6140
                                ;6141          ;--------------------------------;
U 089C, 0918,0CB7,0030,0687,008A,5 ;6142       MEMSCAR_ZLIT24[0],CLEAR FLAG3   ; LOAD ADDRESS OF MME FOR SAVING
                                ;6143
                                ;6144          ;--------------------------------;
U 08A5, 04A4,0036,4033,4647,008A,C ;6145       R[TEMP13]_MEMSCR,STEPC_2        ; SAVE MME
                                ;6146
                                ;6147  =0      ;0-------------------------------;
                                ;6148          M[TEMP5]_ZLIT0[0D],DEC STEPC,   ; TYPE <CR><LF> STEPC=1.
U 08AC, 059E,5C37,0030,6847,049F,4 ;6149       PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;6150
                                ;6151          ;1-------------------------------; SETUP TO TYPE PC.
U 08AD, 0425,A592,4031,C047,0088,C ;6152       R[TEMP7]_M[PC],CLEAR MM.NOINT   ; CN.TYPE.NUM WANTS # IN TEMP7
                                ;6153
                                ;6154  =00     ;00------------------------------;
                                ;6155          M[TEMP10]_ZLIT0[8],             ; SETUP # OF DIGITS TO TYPE.
U 088C, 0986,AC37,0030,4047,04A2,9 ;6156       PUSH,NEXT/CN.TYPE.NUMBER        ;
                                ;6157
                                ;6158          ;01------------------------------;
                                ;6159          M[TEMP5]_ZLIT0[20],STEPC_2,     ; TYPE 2 SPACES.
U 088D, 01A6,5C37,0031,0047,049F,4 ;6160       PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;6161
                                ;6162          ;10------------------------------; SETUP TO TYPE HALT CODE.
U 088E, 0884,C3B7,0001,C047,0088,8 ;6163       R[TEMP7]_M[FPDOFFSET].RR.8      ; CN.TYPE.NUM WANTS # IN TEMP7
                                ;6164  =
                                ;6165  =000    ;000-----------------------------;
                                ;6166          M[TEMP10]_CONX(2),              ; SETUP # OF DIGITS TO TYPE.
U 0888, 0486,A737,0010,0047,04A2,9 ;6167       PUSH,NEXT/CN.TYPE.NUMBER        ;
                                ;6168
                                ;6169          ;001-----------------------------;
                                ;6170          RNUM_D_ZLIT0[0A],               ; CLEAR THE TEMPS
U 0889, 0581,DC35,2030,5047,0497,1 ;6171       PUSH,NEXT/CN.CLEAR.TEMPS        ;
                                ;6172
                                ;6173          ;001-----------------------------;
U 088A, 0986,9C37,0030,4047,008A,8 ;6174       M[TEMP9]_ZLIT0[8]               ; SETUP DEFAULT SIZE TO LONG
                                ;6175  =
                                ;6176          ;--------------------------------;
                                ;6177          R[TEMP9]_M[TEMP9],              ; SETUP SAVED SIZE TO LONG
U 08AB, 0484,9592,4032,4047,008B,C ;6178       NEXT/CN.SAVE.PSL                ;
```

```
                                ;6179  .TOC "  Console Command Parser          :  Type Console Prompt, Decode Command Chara
                                ;6180
                                ;6181  ;*****************************************************************************
                                ;6182  ;       Entry Points    CN.PROMPT
                                ;6183  ;
                                ;6184  ;       Input           FPDOFFSET       Halt code
                                ;6185  ;
                                ;6186  ;       Output          TEMP7           Cleared at start of new command line.
                                ;6187  ;                                       Used to build checksum for X command.
                                ;6188  ;                       TEMP0-TEMP3     Cleared
                                ;6189  ;                       FPDOFFSET       80000000
                                ;6190  ;
                                ;6191  ;       Resources       RNUM            If booting 1 means do micro verify
                                ;6192  ;                                       also used in clearing temps
                                ;6193  ;                       (see Console Resource Usage)
                                ;6194  ;                       TEMP5, TEMP6
                                ;6195  ;
                                ;6196  ;       Subroutines/External Labels
                                ;6197  ;                       CN.TYPE.CHAR
                                ;6198  ;                       CN.PARSE.COMMAND
                                ;6199  ;                       CN.GET.NEXT
                                ;6200  ;                       CN.CLEAR.TEMPS
                                ;6201  ;                       CN.TYPE.PC
                                ;6202  ;                       CN.DO.COMMAND
                                ;6203  ;                       BO.BOOT1
                                ;6204  ;                       BO.ACTION_SWITCH
                                ;6205  ;                       BO.RESTART_HALT
                                ;6206  ;
                                ;6207  ;       This section types the console input prompt <CR><LF>'>>>' and
                                ;6208  ;       then checks the first character of the command line.
                                ;6209  ;       If a legal command character is typed TEMP6 is setup for that command
                                ;6210  ;       and control is passed to CN.PARSE.COMMAND which finishes parsing the
                                ;6211  ;       command line.  <CR> and ^P are also recognized and the proper action is
                                ;6212  ;       taken (a new prompt or re-initialization of the console respectively).
                                ;6213  ;       If the entire command line is rubbed out, CN.PARSE.COMMAND will return
                                ;6214  ;       and processing continues at CN.NEW.LINE as if a prompt had just been
                                ;6215  ;       typed.
                                ;6216  ;*****************************************************************************
                                ;6217
                                ;6218  8AF:    ;*********FORCE ADDRESS*********;
                                ;6219  CN.PROMPT:
                                ;6220          ;-------------------------------; SHUT OUT ALL INTERRUPTS AND GIVE
U 08AF, 0120,0D37,0030,F807,008C,0  ;6221          PSL_ZLIT16[1F],CLEAR MM.NOINT   ; KERNAL PRIVILEDGE FOR MEMORY ACCESS.
                                ;6222
                                ;6223  =0       ;0-------------------------------;
                                ;6224          M[TEMP5]_ZLIT0[0D],             ; TYPE <CR><LF>
U 08C0, 0586,5C37,0030,6847,049F,4  ;6225          PUSH,NEXT/CN.TYPE.CHAR          ; CHANGE ACCESS MODE TO KERNAL
                                ;6226
                                ;6227          ;1-------------------------------;
                                ;6228          WB_M[FPDOFFSET].XOR.ZLIT0[6],   ;******NOP LEFT TO AVOID ROM CHANGE**
U 08C1, 0580,CC13,4A30,3047,00FE,7  ;6229          WX.E0.0?                        ;
                                ;6230
                                ;6231  0FE7:   ;*********FORCE ADDRESS*********;CONSTRAIN TO BLOCK PREVIOUS BRANCH
                                ;6232          CRAR_ZLIT16[0C0],               ;LOAD CRAR FOR CLEARING HALT PENDING.
U 0FE7, 0580,0D37,06B6,0247,008D,0  ;6233          STACK FLAG?                     ;CHECK IF ATTEMPT RESTART/BOOT
```

```
                                        ;6234  =0      ;0------------------------------; KEEP MEMSCAR POINTING TO MME FOR
                                        ;6235          MEMSCAR_ZLIT24[0],              ; SWITCHING BETWEEN VIRTUAL AND PHYISCAL
U 08D0, 0980,CCB7,0030,0687,0089,7      ;6236          NEXT/CN.PROMPT.10               ; FOR EXAMINES AND DEPOSITS.
                                        ;6237
                                        ;6238          ;1------------------------------; ATTEMPT RESTART/BOOT
                                        ;6239          RNUM_[1],CLEAR STACK FLAG,      ; CHECK BOOT ACTION SWITCH
U 08D1, 0529,DC37,0D30,0847,0089,4      ;6240          FPS1?                           ; DO MICRO VERIFY, IF BOOTING
                                        ;6241  =0000
                                        ;6242  =0100    ;0100----------------------------; RESTART/BOOT
                                        ;6243          CONREGS_ZLIT16[40],             ; CLEAR HALT AND HALT PENDING.
U 0894, 0D80,0D37,0032,02C7,0080,4      ;6244          NEXT/BO.ACTION_SWITCH           ;
                                        ;6245
                                        ;6246          ;0101----------------------------; RESTART/HALT
                                        ;6247          CONREGS_ZLIT16[40],             ; CLEAR HALT AND HALT PENDING.
U 0895, 0580,0D37,0032,02C7,0080,5      ;6248          NEXT/BO.RESTART_HALT            ;
                                        ;6249
                                        ;6250          ;0110----------------------------; BOOT
U 0896, 0884,05B7,0035,4047,0080,6      ;6251          R[R5]_0,NEXT/BO.BOOT            ; CLEAR BOOT CONTROL FLAGS
                                        ;6252
                                        ;6253  CN.PROMPT.10:
                                        ;6254          ;0111----------------------------; HALT
                                        ;6255          M[TEMP5]_ZLIT0[3E],STEPC_2,     ; TYPE PROMPT, > IS TYPED TWICE BY
U 0897, 0DA6,5C37,0031,F047,049F,4      ;6256          PUSH,NEXT/CN.TYPE.CHAR          ; CN.TYPE.CHAR AND ONCE BY CN.GET.NEXT.
                                        ;6257
                                        ;6258  CN.NEW.LINE:
                                        ;6259          ;1000----------------------------; TYPE CHAR IN TEMP5 AND CLEAR CHECKSUM
                                        ;6260          M[TEMP7]_Q_R[ZERO],             ; FOR APT AND CLEAR Q FOR DEFAULT ADDR.
U 0898, 0486,75BE,503D,8047,049E,A      ;6261          PUSH,NEXT/CN.GET.NEXT           ; GET NEXT CHAR FROM CONSOLE INTO TEMP5.
                                        ;6262
                                        ;6263          ;1001----------------------------;
                                        ;6264          M[TEMP5]_ZLIT0[7],              ; NOTHING TO RUBOUT, RING BELL
U 0899, 0586,5C37,0030,3847,0089,8      ;6265          NEXT/CN.NEW.LINE                ;
                                        ;6266
                                        ;6267          ;1010----------------------------;
                                        ;6268          CLEAR FLAG0,                    ; CLEAR FLAG TO START WITH BUF0
                                        ;6269          RNUM_D_ZLIT0[3],                ; CLEAR BUFFERS AND BUFFER COUNTERS
U 089A, 0D01,DC35,2030,1847,0497,1      ;6270          PUSH,NEXT/CN.CLEAR.TEMPS        ;
                                        ;6271
                                        ;6272          ;1011----------------------------; CLEAR HALT CODE AND SET BIT TO
                                        ;6273          M[FPDOFFSET]_ZLIT28[8],          ; REGAIN CONTROL FROM MM AND IANDE.
U 089B, 0586,CC77,0030,4047,0481,F      ;6274          PUSH,NEXT/BO.COLD_START_FLAG     ; CLEAR COLD START FLAG.
                                        ;6275
                                        ;6276  =1110    ;1110----------------------------;
U 089E, 0880,05BE,4033,4607,008A,E      ;6277          MEMSCR_R[TEMP13]                ; RESTORE MME BEFORE EACH COMMAND
                                        ;6278  =
```

```
                                ;6279           ;-------------------------------;
                                ;6280           WB_M[TEMP5].XOR.ZLIT0[45],      ; CHECK FOR 'EXAMINE' COMMAND
U 08AE, 0980,5C13,4A32,2847,0089,0 ;6281        WX.EQ.0?                        ;
                                ;6282
                                ;6283   =00     ;00-----------------------------;
                                ;6284           WB_M[TEMP5].XOR.ZLIT0[44],      ; CHECK FOR 'DEPOSIT' COMMAND
U 0890, 0580,5C13,4A32,2047,008A,8 ;6285        WX.EQ.0?,NEXT/CN.TRY.D          ;
                                ;6286
                                ;6287           ;01-----------------------------;
                                ;6288           M[TEMP6]_OLIT0[108],            ; SETUP SWITCH FLAGS FOR E COMMAND.
U 0891, 0986,6E37,0038,4047,0493,4 ;6289        PUSH,NEXT/CN.PARSE.COMMAND      ; PROCESS AN EXAMINE.
                                ;6290
                                ;6291           ;10-----------------------------;
                                ;6292           M[TEMP5]_ZLIT0[45],             ; SETUP TO ECHO RUBOUT OF 'E'.
U 0892, 0586,5C37,0032,2847,0089,8 ;6293        NEXT/CN.NEW.LINE                ;
                                ;6294   =
                                ;6295   =00
                                ;6296   CN.TRY.D:
                                ;6297           ;00-----------------------------;
                                ;6298           WB_M[TEMP5].XOR.ZLIT0[48],      ; CHECK FOR 'HALT' COMMAND
U 08A8, 0180,5C13,4A32,4047,008C,C ;6299        WX.EQ.0?,NEXT/CN.TRY.H          ;
                                ;6300
                                ;6301           ;01-----------------------------;
                                ;6302           M[TEMP6]_OLIT0[110],            ; SETUP SWITCH FLAGS FOR D COMMAND.
U 08A9, 0986,6F37,0038,8047,0493,4 ;6303        PUSH,NEXT/CN.PARSE.COMMAND      ; PROCESS A DEPOSIT.
                                ;6304
                                ;6305           ;10-----------------------------;
                                ;6306           M[TEMP5]_ZLIT0[44],             ; SETUP TO ECHO RUBOUT OF 'D'.
U 08AA, 0186,5C37,0032,2047,0089,8 ;6307        NEXT/CN.NEW.LINE                ;
                                ;6308   =
                                ;6309   =00
                                ;6310   CN.TRY.H:
                                ;6311           ;00-----------------------------;
                                ;6312           WB_M[TEMP5].XOR.ZLIT0[4E],      ; CHECK FOR 'NEXT' COMMAND
U 08CC, 0580,5C13,4A32,7047,008D,C ;6313        WX.EQ.0?,NEXT/CN.TRY.N          ;
                                ;6314
                                ;6315           ;01-----------------------------;
                                ;6316           M[TEMP6]_OLIT0[12F],            ; SETUP SWITCH FLAGS FOR H COMMAND.
U 08CD, 0586,6E37,0039,7847,0493,4 ;6317        PUSH,NEXT/CN.PARSE.COMMAND      ; PROCESS A HALT.
                                ;6318
                                ;6319           ;10-----------------------------;
                                ;6320           M[TEMP5]_ZLIT0[48],             ; SETUP TO ECHO RUBOUT OF 'H'.
U 08CE, 0D86,5C37,0032,4047,0089,8 ;6321        NEXT/CN.NEW.LINE                ;
                                ;6322   ;
```

```
                                ;6323  =00
                                ;6324  CN.TRY.N:
                                ;6325          ;00--------------------------------;
                                ;6326          WB_M[TEMP5].XOR.ZLIT0[53],       ; CHECK FOR 'START' COMMAND
U 08DC, 0980,5C13,4A32,9847,0090,0 ;6327       WX.EQ.0?,NEXT/CN.TRY.S           ;
                                ;6328
                                ;6329          ;01--------------------------------;
                                ;6330          M[TEMP6] OLIT0[13F],             ; SETUP SWITCH FLAGS FOR N COMMAND.
U 08DD, 0186,6E37,0039,F847,0493,4 ;6331       PUSH,NEXT/CN.PARSE.COMMAND       ; PROCESS A NEXT.
                                ;6332
                                ;6333          ;10--------------------------------;
                                ;6334          M[TEMP5] ZLIT0[4E],              ; SETUP TO ECHO RUBOUT OF 'N'.
U 08DE, 0186,5C37,0032,7047,0089,8 ;6335       NEXT/CN.NEW.LINE                 ;
                                ;6336  =
                                ;6337  =00
                                ;6338  CN.TRY.S:
                                ;6339          ;00--------------------------------;
                                ;6340          WB_M[TEMP5].XOR.ZLIT0[43],       ; CHECK FOR 'CONTINUE' COMMAND
U 0900, 0580,5C13,4A32,1847,0092,0 ;6341       WX.EQ.0?,NEXT/CN.TRY.C           ;
                                ;6342
                                ;6343          ;01--------------------------------;
                                ;6344          M[TEMP6] OLIT0[157],             ; SETUP SWITCH FLAGS FOR S COMMAND.
U 0901, 0586,6E37,003A,B847,0493,4 ;6345       PUSH,NEXT/CN.PARSE.COMMAND       ; PROCESS A START.
                                ;6346
                                ;6347          ;10--------------------------------;
                                ;6348          M[TEMP5] ZLIT0[53],              ; SETUP TO ECHO RUBOUT OF 'S'.
U 0902, 0D86,5C37,0032,9847,0089,8 ;6349       NEXT/CN.NEW.LINE                 ;
                                ;6350  =
                                ;6351  =00
                                ;6352  CN.TRY.C:
                                ;6353          ;00--------------------------------;
                                ;6354          WB_M[TEMP5].XOR.ZLIT0[54],       ; CHECK FOR 'TEST' COMMAND
U 0920, 0180,5C13,4A32,A047,0092,4 ;6355       WX.EQ.0?,NEXT/CN.TRY.T           ;
                                ;6356
                                ;6357          ;01--------------------------------;
                                ;6358          M[TEMP6] OLIT0[14F],             ; SETUP SWITCH FLAGS FOR C COMMAND.
U 0921, 0586,6E37,003A,7847,0493,4 ;6359       PUSH,NEXT/CN.PARSE.COMMAND       ; PROCESS A CONTINUE.
                                ;6360
                                ;6361          ;10--------------------------------;
                                ;6362          M[TEMP5] ZLIT0[43],              ; SETUP TO ECHO RUBOUT OF 'C'.
U 0922, 0986,5C37,0032,1847,0089,8 ;6363       NEXT/CN.NEW.LINE                 ;
                                ;6364  =
```

```
                                ;6365  =00
                                ;6366  CN.TRY.T:
                                ;6367          ;00------------------------------;
                                ;6368          WB_M[TEMP5].XOR.ZLIT0[42],      ; CHECK FOR 'BOOT' COMMAND
U 0924, 0980,5C13,4A32,1047,0092,8 ;6369       WX.EQ.0?,NEXT/CN.TRY.B          ;
                                ;6370
                                ;6371          ;01------------------------------;
                                ;6372          M[TEMP6]_OLIT0[1DF],            ; SETUP SWITCH FLAGS FOR T COMMAND.
U 0925, 0586,6E37,003E,F847,0493,4 ;6373       PUSH,NEXT/CN.PARSE.COMMAND      ; PROCESS A TEST.
                                ;6374
                                ;6375          ;10------------------------------;
                                ;6376          M[TEMP5]_ZLIT0[54],             ; SETUP TO ECHO RUBOUT OF 'T'.
U 0926, 0586,5C37,0032,A047,0089,8 ;6377       NEXT/CN.NEW.LINE                ;
                                ;6378  =
                                ;6379  =00
                                ;6380  CN.TRY.B:
                                ;6381          ;00------------------------------;
                                ;6382          WB_M[TEMP5].XOR.ZLIT0[58],      ; CHECK FOR 'X' COMMAND
U 0928, 0580,5C13,4A32,C047,0092,C ;6383       WX.EQ.0?,NEXT/CN.TRY.X          ;
                                ;6384
                                ;6385          ;01------------------------------;
                                ;6386          M[TEMP6]_ZLIT0[07F],SET FLAG3,  ; SETUP SWITCH FLAGS FOR B COMMAND.
U 0929, 055E,6C37,0033,F847,0493,4 ;6387       PUSH,NEXT/CN.PARSE.COMMAND      ; PROCESS A BOOT.
                                ;6388
                                ;6389          ;10------------------------------;
                                ;6390          M[TEMP5]_ZLIT0[42],             ; SETUP TO ECHO RUBOUT OF 'B'.
U 092A, 0D86,5C37,0032,1047,0089,8 ;6391       NEXT/CN.NEW.LINE                ;
                                ;6392  =
                                ;6393  =00
                                ;6394  CN.TRY.X:
                                ;6395          ;00------------------------------;
                                ;6396          WB_M[TEMP5].XOR.ZLIT0[49],      ; CHECK FOR 'I'
U 092C, 0D80,5C13,4A32,4847,0093,0 ;6397       WX.EQ.0?,NEXT/CN.TRY.I          ;
                                ;6398
                                ;6399          ;01------------------------------;
                                ;6400          M[TEMP6]_OLIT0[1A3],            ; SETUP SWITCH FLAGS FOR X COMMAND.
U 092D, 0D86,6E37,003D,1847,0493,4 ;6401       PUSH,NEXT/CN.PARSE.COMMAND      ; PROCESS A TRANSFER.
                                ;6402
                                ;6403          ;10------------------------------;
                                ;6404          M[TEMP5]_ZLIT0[58],             ; SETUP TO ECHO RUBOUT OF 'X'.
U 092E, 0986,5C37,0032,C047,0089,8 ;6405       NEXT/CN.NEW.LINE                ;
                                ;6406  =
```

```
                                ;6407  =00
                                ;6408  CN.TRY.I:
                                ;6409          ;00-------------------------------;
                                ;6410          WB_M[TEMP5].XOR.ZLIT0[0D],      ; CHECK FOR <CR>
U 0930, 0180,5C13,4A30,6847,008D,6 ;6411       WX.EQ.0?,NEXT/CN.TRY.CR         ;
                                ;6412
                                ;6413          ;01-------------------------------;
                                ;6414          M[TEMP6]_OLIT0[1BF],            ; SETUP SWITCH FLAGS FOR I COMMAND.
U 0931, 0586,6E37,003D,F847,0493,4 ;6415       PUSH,NEXT/CN.PARSE.COMMAND      ; PROCESS AN INITIALIZE.
                                ;6416
                                ;6417          ;10-------------------------------;
                                ;6418          M[TEMP5]_ZLIT0[49],             ; SETUP TO ECHO RUBOUT OF 'I'.
U 0932, 0986,5C37,0032,4847,0089,8 ;6419       NEXT/CN.NEW.LINE                ;
                                ;6420  =
                                ;6421  =0
                                ;6422  CN.TRY.CR:
                                ;6423          ;0--------------------------------;
                                ;6424          WB_M[TEMP5].XOR.ZLIT0[10],      ; CHECK FOR ^P
U 08D6, 0D80,5C13,4A30,8047,0097,4 ;6425       WX.EQ.0?,NEXT/CN.TRY.CNTRL.P    ;
                                ;6426
                                ;6427          ;1--------------------------------;
U 08D7, 0C80,0036,4030,0047,008A,F ;6428       NEXT/CN.PROMPT                  ; TYPE PROMPT IN RESPONSE TO A <CR>.
                                ;6429  =0
                                ;6430  CN.TRY.CNTRL.P:
                                ;6431          ;0--------------------------------;
                                ;6432          M[TEMP5]_ZLIT0[7],              ; UNRECOGNIZED COMMAND RING BELL.
U 0974, 0586,5C37,0030,3847,0089,8 ;6433       NEXT/CN.NEW.LINE                ;
                                ;6434
                                ;6435          ;1--------------------------------;
                                ;6436          PC_R[RTMPGPR]+CONX(2),          ; RESTORE PC
U 0975, 0080,073D,0017,C487,0088,4 ;6437       NEXT/CN.TYPE.PC                 ; RESTART CONSOLE.
```

```
                                ;6438  .TOC "  Console Command Parser          : Command Parser"
                                ;6439
                                ;6440  ;*************************************************************************
                                ;6441  ;       Entry Points    CN.PARSE.COMMAND
                                ;6442  ;
                                ;6443  ;       Resources       TEMP0, TEMP1, TEMP5, TEMP6, FLAG0, FLAG3
                                ;6444  ;
                                ;6445  ;       Subroutines/External Labels
                                ;6446  ;                       CN.GET.NEXT
                                ;6447  ;                       CN.DO.COMMAND
                                ;6448  ;                       CN.GET.SWITCH
                                ;6449  ;                       CN.GET.BOOT.DEV
                                ;6450  ;                       CN.GET.NUM.RUBOUT
                                ;6451  ;*************************************************************************
                                ;6452
                                ;6453  =00
                                ;6454  CN.PARSE.COMMAND:
                                ;6455          ;00-----------------------------;
U 0934, 0480,0036,4030,0047,049E,A ;6456        PUSH,NEXT/CN.GET.NEXT           ; CONTINUE PARSING E COMMAND.
                                ;6457
                                ;6458          ;01-----------------------------;
U 0935, 0880,0036,40B0,0047,0000,1 ;6459        RETURN [+1]                     ;
                                ;6460
                                ;6461  CN.PARSE.RECURSE:
                                ;6462          ;10-----------------------------;
                                ;6463          WB_M[TEMP5].XOR.ZLIT0[2F],      ; IS NEXT CHAR A '/'.
U 0936, 0D80,5C13,4A71,7847,0097,6 ;6464        WX.NE.0?                        ;
                                ;6465  =
                                ;6466  =0      ;0-----------------------------;
U 0976, 0C80,0036,4030,0047,0097,C ;6467        NEXT/CN.PARSE.SWITCHES          ; EVAULUATE THE SWITCH
                                ;6468
                                ;6469          ;1-----------------------------;
                                ;6470          WB_M[TEMP5].XOR.ZLIT0[20],      ; IS NEXT CHAR A ' '.
U 0977, 0980,5C13,4A71,0047,0097,8 ;6471        WX.NE.0?                        ;
                                ;6472
                                ;6473  =0      ;0-----------------------------; NEXT CHAR IS A ' ', CHECK FOR BOOT
U 0978, 0080,0036,44F0,0047,0097,E ;6474        FLAG3?,NEXT/CN.BOOT.DEV         ; BOOT NEEDS SPECIAL HANDLING.
                                ;6475
                                ;6476          ;1-----------------------------;
                                ;6477          WB_M[TEMP5].XOR.ZLIT0[0D],      ; IS NEXT CHAR A '<CR>'.
U 0979, 0D80,5C13,4A70,6847,0097,A ;6478        WX.NE.0?                        ;
                                ;6479
                                ;6480  =0      ;0-----------------------------; <CR> TYPED, END OF LINE
                                ;6481          SET MM.NOINT,                   ; SET TU BLOCK INTERRUPTS
                                ;6482          WB_M[TEMP6].RR.4,               ; GO PROCESS THE COMMAND, TEMP6 HAS
U 097A, 0860,6277,0231,8047,008B,0 ;6483        WB<5-0>?,NEXT/CN.DO.COMMAND     ; THE COMMAND NUMBER
                                ;6484
                                ;6485          ;1-----------------------------;
                                ;6486          M[TEMP5]_ZLIT0[7],              ; UNRECOGNIZED CHARACTER, RING BELL
U 097B, 0D86,5C37,0030,3847,0093,4 ;6487        NEXT/CN.PARSE.COMMAND           ;
```

```
                                ;6488  =0
                                ;6489  CN.PARSE.SWITCHES:
                                ;6490          ;0------------------------------;
U 097C, 0C80,0036,4030,0047,0494,8  ;6491          PUSH,NEXT/CN.GET.SWITCH       ; JUMP TO COMMON SWITCH ROUTINE.
                                ;6492
                                ;6493          ;1------------------------------;
U 097D, 0C80,0036,4030,0047,0093,4  ;6494          NEXT/CN.PARSE.COMMAND         ; CATCH RUBOUT RETURN FROM CN.GET.SWITCH
                                ;6495  =0
                                ;6496  CN.BOOT.DEV:
                                ;6497          ;0------------------------------; NOT BOOT COMMAND, INPUT ADDRESS.
                                ;6498          WB_M[TEMP6].AND.CONX(4),      ; CHECK IF ADDRESS HAS ALREADY BEEN SET.
U 097E, 0880,6712,0A20,0047,008A,0  ;6499          WX.EQ.0?,NEXT/CN.PARSE.ADDRESS ;
                                ;6500
                                ;6501          ;1------------------------------;
                                ;6502          M[TEMP0] ZLIT0[3],            ; SETUP TO READ IN DEV NAME
U 097F, 0586,0C37,0030,1847,008E,8  ;6503          NEXT/CN.GET.BOOT.DEV          ;
                                ;6504  =000
                                ;6505  CN.PARSE.ADDRESS:
                                ;6506          ;000-----------------------------;
                                ;6507          WB_M[TEMP6].AND.ZLIT0[8],     ; CHECK IF DEPOSIT NEEDS DATA STILL.
U 08A0, 0D80,6C12,0A30,4047,008E,0  ;6508          WX.EQ.0?,NEXT/CN.PARSE.DATA   ;
                                ;6509
                                ;6510  CN.PARSE.NUMBER:
                                ;6511          ;001------------------------------;
                                ;6512          M[TEMP6] MB.OR.CONX(4),       ; INDICATE ADDRESS HAS BEEN SPECIFIED.
U 08A1, 0486,6712,4020,0047,0491,0  ;6513          PUSH,NEXT/CN.GET.NUMBER       ; INPUT HEX NUMBER FROM CONSOLE.
                                ;6514
                                ;6515          ;010------------------------------;
                                ;6516          M[TEMP6] MB.ANDNOT.CONX(4),   ; ALLOW NEW ADDRESS TO BE TYPED.
U 08A2, 0886,6713,8020,0047,008A,6  ;6517          NEXT/CN.PARSE.RUBOUT.SPACE    ;
                                ;6518
                                ;6519          ;011------------------------------;
U 08A3, 0C80,0036,4030,0047,0493,6  ;6520          PUSH,NEXT/CN.PARSE.RECURSE    ; CONTINUE PARSING E RECURSIVELY.
                                ;6521
                                ;6522          ;100------------------------------;
U 08A4, 0480,0036,4030,0047,049E,6  ;6523          PUSH,NEXT/CN.GET.NUM.RUBOUT   ; RUBOUT LAST DIGIT OF ADDRESS
                                ;6524  =110
                                ;6525  CN.PARSE.RUBOUT.SPACE:
                                ;6526          ;110------------------------------;
                                ;6527          M[TEMP5] ZLIT0[20],           ; SETUP TO ECHO RUBOUT OF ' '.
U 08A6, 0186,5C37,0031,0047,0093,4  ;6528          NEXT/CN.PARSE.COMMAND         ; CONTINUE PARSING E COMMAND
                                ;6529
                                ;6530          ;111------------------------------;
U 08A7, 0480,0036,4030,0047,008A,1  ;6531          NEXT/CN.PARSE.NUMBER          ; CONTINUE GETTING NUMBER
                                ;6532  =000
                                ;6533  CN.PARSE.DATA:
                                ;6534          ;000-----------------------------;
                                ;6535          M[TEMP5] ZLIT0[7],            ; EXAMINE OR DATA HAS ALREADY BEEN SET,
U 08E0, 0D86,5C37,0030,3847,0093,4  ;6536          NEXT/CN.PARSE.COMMAND         ; RING BELL.
                                ;6537
                                ;6538  CN.PARSE.GET.DATA:
                                ;6539          ;001------------------------------;
                                ;6540          SET FLAG0,                    ; SET FLAG TO INDICATE USE BUF1.
                                ;6541          M[TEMP6] MB.OR.ZLIT0[8],      ; INDICATE DATA HAS BEEN SPECIFIED.
U 08E1, 0946,6C12,4030,4047,0491,0  ;6542          PUSH,NEXT/CN.GET.NUMBER       ; INPUT HEX NUMBER FROM CONSOLE.
```

```
                                        ;6543           ;010----------------------------;
                                        ;6544           M[TEMP6] MB.ANDNOT.ZLIT0[8],    ; ALLOW NEW DATA TO BE TYPED.
U 08E2, 0D86,6C13,8030,4047,008E,6      ;6545           NEXT/CN.PARSE.RUBOUT.DATA       ;
                                        ;6546
                                        ;6547           ;011----------------------------;
U 08E3, 0C80,0036,4030,0047,0493,6      ;6548           PUSH,NEXT/CN.PARSE.RECURSE      ; CONTINUE PARSING D RECURSIVELY.
                                        ;6549
                                        ;6550           ;100----------------------------;
                                        ;6551           FLAG0?,                         ; BRANCH OF BUF0 OR BUF1.
U 08E4, 0080,0036,4430,0047,049E,6      ;6552           PUSH,NEXT/CN.GET.NUM.RUBOUT     ; RUBOUT LAST DIGIT OF DATA
                                        ;6553   =110
                                        ;6554   CN.PARSE.RUBOUT.DATA:
                                        ;6555           ;110----------------------------; CLEAR FLAG0 TO INDICATE BUF0
                                        ;6556           M[TEMP5] ZLIT0[20],CLEAR FLAG0, ; SETUP TO ECHO RUBOUT OF ' '.
U 08E6, 0906,5C37,0031,0047,0093,4      ;6557           NEXT/CN.PARSE.COMMAND           ; CONTINUE PARSING D COMMAND
                                        ;6558
                                        ;6559           ;111----------------------------;
U 08E7, 0C80,0036,4030,0047,008E,1      ;6560           NEXT/CN.PARSE.GET.DATA          ; CONTINUE GETTING NUMBER
```

```
                                    ;6561  .TOC "  Console Command Parser          :       Parse Device for Boot Command"
                                    ;6562
                                    ;6563  ;*************************************************************************
                                    ;6564  ;       Entry Point     CN.GET.BOOT.DEV
                                    ;6565  ;
                                    ;6566  ;       Resources       TEMP0, TEMP1, FLAG3
                                    ;6567  ;
                                    ;6568  ;       Subroutines/External Labels
                                    ;6569  ;                       CN.GET.NEXT
                                    ;6570  ;                       CN.PARSE.CONT
                                    ;6571  ;
                                    ;6572  ;       This routine will save the next for characters from the terminal
                                    ;6573  ;       in BUF0.  The only characters that are not put in the buffer are
                                    ;6574  ;       ^U and DELETE which have there usual function. That means control
                                    ;6575  ;       characters and <CR> will just be put in the buffer.
                                    ;6576  ;*************************************************************************
                                    ;6577  =000
                                    ;6578  CN.GET.BOOT.DEV:
                                    ;6579          ;000-------------------------------;
                                    ;6580          M[TEMP1]_MB.RL.8,                 ; PREPARE FOR NEXT BYTE.
U 08E8, 0086,13B7,0020,0047,049E,A  ;6581          PUSH,NEXT/CN.GET.NEXT             ;
                                    ;6582
                                    ;6583  CN.GET.BOOT.DEV.05:
                                    ;6584          ;001-------------------------------;
                                    ;6585          M[TEMP0]_MB+1,                    ; RUBOUT LAST CHARACTER
                                    ;6586          WB<1-0>.NE.0?,                    ; CHECK IF NO CHARACTERS TO RUBOUT
U 08E9, 0486,0081,03BD,8047,0898,0 347*  ;6587          NEXT/CN.GET.BOOT.DEV.20           ;
                                    ;6588
                                    ;6589          ;010-------------------------------;
                                    ;6590          R[TEMP1].SIZ_M[TEMP5],            ; PUT CHARACTER IN BUFFER.
                                    ;6591          SIZE[BYTE],                       ;
U 08EA, 0082,5592,4000,4047,008B,F  ;6592          NEXT/CN.GET.BOOT.DEV.10           ;
                                    ;6593
                                    ;6594          ;011-------------------------------;
                                    ;6595          CLEAR FLAG3,                      ; INDICATE BOOT DEVICE HAS BEEN READ
U 08EB, 0C18,0036,4030,0047,0493,4  ;6596          PUSH,NEXT/CN.PARSE.COMMAND        ; CONTINUE PARSING COMMAND.
                                    ;6597
                                    ;6598          ;100-------------------------------;
                                    ;6599          SET FLAG3,                        ; INDICATE STILL READING BOOT DEVICE
U 08EC, 005E,13B7,0020,0047,008B,6  ;6600          M[TEMP1]_MB.RL.8                  ; PREPARE TO RUBOUT LAST CHARACTER.
                                    ;6601  =
                                    ;6602          ;-------------------------------;
                                    ;6603          M[TEMP0]_MB+1,                    ; RUBOUT LAST CHARACTER
U 08B6, 0086,0081,003D,8047,0098,1  ;6604          NEXT/CN.GET.BOOT.DEV.25           ;
```

```
                                        ;6605  CN.GET.BOOT.DEV.10:
                                        ;6606          ;--------------------------------;
                                        ;6607          M[TEMP0]_MB-CONX(1),             ; DECREMENT COUNT OF CHARACTER TO FETCH
U 08BF, 0086,0710,06C0,0047,088E,8 376* ;6608          WB<31-30>?,NEXT/CN.GET.BOOT.DEV  ;
                                        ;6609  =0
                                        ;6610  CN.GET.BOOT.DEV.20:
                                        ;6611          ;0--------------------------------;
                                        ;6612          M[TEMP5]_ZLIT0[20],              ; ENTIRE DEVICE NAME RUBBED OUT, PREPARE
U 0980, 0186,5C37,0031,0047,0093,4      ;6613          NEXT/CN.PARSE.COMMAND            ; TO RUBOUT ' '.
                                        ;6614
                                        ;6615  CN.GET.BOOT.DEV.25:
                                        ;6616          ;1--------------------------------;
U 0981, 0886,13B7,0010,0047,008C,A      ;6617          M[TEMP1]_MB.RR.16                ;
                                        ;6618
                                        ;6619          ;--------------------------------;
                                        ;6620          M[TEMP5]_R[TEMP1].RR.24,         ; LOAD CHARACTER TO RUBOUT
U 08CA, 0C86,52B7,0020,4047,008E,8      ;6621          NEXT/CN.GET.BOOT.DEV             ;
```

```
                                ;6622  .TOC "  Console Command Parser          : Console Command Dispatch Table"
                                ;6623
                                ;6624  ;*****************************************************************************
                                ;6625  ;       Entry Point     CN.DO.COMMAND
                                ;6626  ;
                                ;6627  ;       Resources       TEMP1           Temporary for examine PSL
                                ;6628  ;                       TEMP3           Temporary for NEXT and CONTINUE
                                ;6629  ;                       TEMP0, MTEMP8, RTMPGPR, FLAG0, FLAG2, STEPC
                                ;6630  ;       Subroutines/External Labels
                                ;6631  ;               CN.DO.EXAMINE
                                ;6632  ;                       CN.DO.DEPOSIT
                                ;6633  ;                       CN.TYPE.PC
                                ;6634  ;                       CN.DO.CONTINUE
                                ;6635  ;                       CN.DO.START
                                ;6636  ;                       CN.DO.BOOT
                                ;6637  ;                       CN.E.PSL
                                ;6638  ;                       CN.PROMPT
                                ;6639  ;                       CN.DO.X
                                ;6640  ;                       IN.INIT
                                ;6641  ;                       MV.TEST
                                ;6642  ;*****************************************************************************
                                ;6643
                                ;6644  =110000
                                ;6645  CN.DO.COMMAND:
                                ;6646          ;110000-------------------------; EXAMINE
                                ;6647          WB_M[TEMP0],WX.NE.0?,           ; CHECK IF NEW ADDRESS WAS TYPED
U 08B0, 0480,0592,4A70,0047,0098,2 ;6648       NEXT/CN.DO.EXAMINE              ;
                                ;6649
                                ;6650          ;110001-------------------------; DEPOSIT
                                ;6651          MEMSCR_M[TEMP8].RL.24,          ; SET MME ACCORDING TO SPACE SWITCH
                                ;6652          FLAG0?,                         ; CHECK IF DATA HAS BEEN SPECIFIED.
U 08B1, 0080,83B7,0400,0607,0098,6 ;6653       NEXT/CN.DO.DEPOSIT              ;
                                ;6654
                                ;6655          ;110010-------------------------; HALT
                                ;6656          PC_R[RTMPGPR]+CONX(2),          ; RESTORE PC FOR TYPE OUT.
U 08B2, 0080,073D,0017,C487,008A,F ;6657       NEXT/CN.PROMPT                  ;
                                ;6658
                                ;6659          ;110011-------------------------; NEXT
                                ;6660          M[TEMP3]_ZLIT16[0C0],           ; SETUP TO CLEAR HALT AND SET HALT PEND
                                ;6661          CLEAR FLAG2,                    ; CLEAR FLAG2 TO AVOID CONFLICT WITH APT
U 08B3, 0116,3D37,0036,0047,0094,F ;6662       NEXT/CN.DO.CONTINUE             ; JOIN CONTINUE
                                ;6663
                                ;6664          ;110100-------------------------; CONTINUE
                                ;6665          M[TEMP3]_ZLIT16[40],            ; SETUP TO CLEAR HALT AND HALT PENDING
                                ;6666          CLEAR FLAG2,                    ; CLEAR FLAG2 TO AVOID CONFLICT WITH APT
U 08B4, 0516,3D37,0032,C047,0094,F ;6667       NEXT/CN.DO.CONTINUE             ;
                                ;6668
                                ;6669          ;110101-------------------------; START
                                ;6670          FLAG0?,SET FLAG0,               ; CHECK IF START ADDRESS HAS BEEN
U 08B5, 0840,0036,4430,0047,0094,0 ;6671       NEXT/CN.DO.START                ; SPECIFIED. SET FLAG0 TO CLEAR MME.
```

```
                                ;6672  =110111
                                ;6673          ;110111-------------------------; BOOT
                                ;6674          FLAG3?,                         ; CHECK IF DEVICE SPECIFIED.
U 08B7, 0880,0036,44F0,0047,0098,C ;6675       NEXT/CN.DO.BOOT                 ; DO BOOT
                                ;6676
                                ;6677          ;111000-------------------------; EXAMINE SAVED COPY OF PSL
                                ;6678          M[TEMP1]_R[TEMP12],             ;
U 08B8, 0886,15BE,4033,0047,0091,A ;6679       NEXT/CN.E.PSL                   ;
                                ;6680
                                ;6681          ;111001-------------------------;
U 08B9, 0080,0036,4430,0047,00BE,C ;6682       FLAG0?,NEXT/CN.DEPOSIT.PSL      ; CHECK IF DATA HAS BEEN SPECIFIED.
                                ;6683
                                ;6684          ;111010-------------------------; X COMMAND FOR APT TRANSFER
                                ;6685          MEMSCAR_M[TEMP0]_0,             ; CLEAR A TEMP TO HOLD THE CHECKSUM.
U 08BA, 0086,05B7,0030,0687,0099,C ;6686       NEXT/CN.DO.X                    ; PREPARE TO CLEAR MME.
                                ;6687
                                ;6688          ;111011-------------------------; INITIALIZE
                                ;6689          SET FLAG2,                      ; SET FLAG TO SAY NOT POWER UP
U 08BB, 0450,0036,4030,0047,0487,9 ;6690       PUSH,NEXT/IN.INIT               ;
                                ;6691
                                ;6692  CN.SAVE.PSL:
                                ;6693          ;111100-------------------------; RETURN FROM INITIALIZE
                                ;6694          R[TEMP12]_PSL,                  ; SAVE PSL FOR NEXT AND CONTINUE
U 08BC, 0014,0036,4033,0087,00FE,1 ;6695       CLEAR FLAG2,NEXT/CN.RESET.STEPC ;
                                ;6696
                                ;6697          ;111101-------------------------; TEST
U 08BD, 0C80,0036,4030,0047,0562,C ;6698       PUSH,NEXT/MV.TEST               ; DO MICRO VERIFY
                                ;6699
                                ;6700          ;111110-------------------------;
                                ;6701          PC_R[RTMPGPR]+CONX(2),          ; RESTORE PC.
U 08BE, 0080,073D,0017,C487,0000,1 ;6702       NEXT/MS.HALT.MICRO              ; RE INITIALIZE THE CONSOLE.
                                ;6703
                                ;6704  0FE1:    ;*********FORCE ADDRESS*********;
                                ;6705  CN.RESET.STEPC:
                                ;6706          ;-------------------------------;
                                ;6707          CONREGS_ZLIT16[90],             ; SET HALT AND HALT PENDING.
                                ;6708          STEPC_2,                        ;
U 0FE1, 01A0,0D37,0034,8207,00E4,F ;6709       NEXT/MP.MTPR.RESET.STEPC        ;
                                ;6710  =
```

```
                                ;6711  .TOC "  Console Command Parser          :        EXAMINE ROUTINE"
                                ;6712
                                ;6713  ;*****************************************************************************
                                ;6714  ;       Entry Point     CN.DO.EXAMINE
                                ;6715  ;
                                ;6716  ;       Output          TEMP7           Address to be typed by CN.TYPE.RESULT
                                ;6717  ;                       RTEMP9          # of digits to type in CN.TYPE.RESULT
                                ;6718  ;                       MTEMP10         Space char to type in CN.TYPE.RESULT
                                ;6719  ;                       MDR             Input IPR # to MP.MFPR and hold result
                                ;6720  ;                                       data to be typed by CN.TYPE.RESULT
                                ;6721  ;
                                ;6722  ;       Resources       MM.TEMP2        Save MDR when converting virtual to phy
                                ;6723  ;                       DREG            Temporary
                                ;6724  ;                       RNUM            Address GPR to be examined
                                ;6725  ;                       LONLIT          Mask undefined bits from PSL
                                ;6726  ;                       TEMP1, TEMP6, MTEMP8, RTEMP8, MTEMP9, RTEMP10
                                ;6727  ;
                                ;6728  ;       Subroutines/External Labels
                                ;6729  ;                       CN.TYPE.RESULT
                                ;6730  ;                       CN.TYPE.CHAR
                                ;6731  ;                       CN.TYPE.DATA
                                ;6732  ;                       MP.MFPR
                                ;6733  ;
                                ;6734  ;       This section processes all examines. MDR is either loaded from memory
                                ;6735  ;       or from a GPR, for IPR's MP.MFPR is called to get the data into MDR.
                                ;6736  ;       If a default address is to be used it is detected here and the new
                                ;6737  ;       default address for the next examine/deposit is also calculated here.
                                ;6738  ;*****************************************************************************
                                ;6739
                                ;6740  =0
                                ;6741  CN.DO.EXAMINE:
                                ;6742          ;0-------------------------------;
U 0982, 0886,15BE,4032,8047,0098,3  ;6743          M[TEMP1]_R[TEMP10]              ; USE ADDRESS FROM PREVIOUS COMMAND.
                                ;6744
                                ;6745          ;1-------------------------------;
U 0983, 0886,75BE,6030,44A7,008C,F  ;6746          VA_D_M[TEMP7]_R[TEMP1]          ; LOAD ADDRESS TO EXAMINE AND SAVE IT
```

```
                                        ;6747           ;--------------------------------; MOVE NEW SPACE SWITCH TO
                                        ;6748           R[TEMP8]_M[TEMP8],               ; SAVED SPACE SWITCH.
U 08CF, 0C84,8592,4272,0047,0093,8      ;6749           WB<1-0>?                         ; BRANCH ON SPACE SWITCH.
                                        ;6750
                                        ;6751   =00     ;00------------------------------;
                                        ;6752           M[TEMP10]_ZLIT0[50],             ; LOAD 'P' FOR CN.TYPE.RESULT
U 0938, 0D86,AC37,0032,8047,008D,A      ;6753           NEXT/CN.EXAMINE                  ;
                                        ;6754
                                        ;6755           ;01------------------------------;
                                        ;6756           R[MM.TEMP2]_M[MDR],               ; SAVE MDR
U 0939, 0885,2592,4039,8047,0491,8      ;6757           PUSH,NEXT/CN.VIRT.TO.PHY         ; GET VIRTUAL ADDRESS INTO TEMP7
                                        ;6758
                                        ;6759           ;10------------------------------;
                                        ;6760           RNUM_R[TEMP1],                   ; LOAD GPR ADDRESS
U 093A, 0081,D5BE,4030,4047,008E,5      ;6761           NEXT/CN.E.GPR                    ; DO EXAMINE OF GPR
                                        ;6762
                                        ;6763           ;11------------------------------;
                                        ;6764           MDR_M[TEMP1],                    ; MOVE IPR # TO MDR FOR MP.MFPR
U 093B, 0480,1002,403D,8467,0090,3      ;6765           NEXT/CN.E.IPR                    ; DO EXAMINE OF IPR
                                        ;6766
                                        ;6767   CN.EXAMINE:
                                        ;6768           ;--------------------------------;
U 08DA, 0C80,83B7,0000,0607,008D,F      ;6769           MEMSCR_M[TEMP8].RL.24            ; SET MME ACCORDING TO SPACE SWITCH
                                        ;6770
                                        ;6771           ;--------------------------------; MOVE NEW SIZE SWITCH TO
                                        ;6772           R[TEMP9]_M[TEMP9],               ; SAVED SIZE SWITCH.
U 08DF, 0884,9592,4232,4047,008C,0      ;6773           WB<5-0>?,NEXT/CN.EXAMINE.10      ; BRANCH ON SIZE.
                                        ;6774
                                        ;6775   =0000
                                        ;6776   CN.EXAMINE.10:
                                        ;6777   =0010   ;0010------------------------------;
                                        ;6778           READ,SIZE[BYTE],                 ;
U 0802, 0C84,0731,0002,8050,008C,3      ;6779           R[TEMP10]_D+CONX(1)               ; UPDATE NEW DEFAULT ADDRESS
                                        ;6780
                                        ;6781           ;0011------------------------------;
                                        ;6782           MDR_M[MDR].RL.24,                ; LEFT JUSTIFY 2 DIGITS TO TYPE.
U 0803, 0081,23B7,0000,0467,008F,8      ;6783           NEXT/CN.TYPE.RESULT              ;
                                        ;6784
                                        ;6785           ;0100------------------------------;
                                        ;6786           READ,SIZE[WORD],                 ;
U 0804, 0884,0731,0012,8050,008C,5      ;6787           R[TEMP10]_D+CONX(2)               ; UPDATE NEW DEFAULT ADDRESS
                                        ;6788
                                        ;6789           ;0101------------------------------;
                                        ;6790           MDR_M[MDR].RL.16,                ; LEFT JUSTIFY 4 DIGITS TO TYPE.
U 0805, 0481,23B7,0010,0467,008F,8      ;6791           NEXT/CN.TYPE.RESULT              ;
                                        ;6792
                                        ;6793   =1000   ;1000------------------------------;
                                        ;6794           READ,SIZE[LONG],                 ;
                                        ;6795           R[TEMP10]_D+CONX(4),              ; UPDATE NEW DEFAULT ADDRESS
U 0808, 0084,0731,0022,8050,008F,8      ;6796           NEXT/CN.TYPE.RESULT              ;
                                        ;6797   =
```

```
                                ;6798  CN.E.GPR:
                                ;6799          ;---------------------------------;
U 08E5, 0884,13F7,0031,C047,008E,E ;6800          R[TEMP7]_M[TEMP1].NIBBLE         ; SAVE GPR ADDRESS CLEARING IGNORED BITS
                                ;6801
                                ;6802          ;---------------------------------;
U 08EE, 0986,AC37,0032,3847,008F,0 ;6803          M[TEMP10]_ZLIT0[47]              ; LOAD 'G' FOR CN.TYPE.RESULT
                                ;6804
                                ;6805          ;---------------------------------;
U 08F0, 0480,05BE,403C,C467,0090,6 ;6806          MDR_R[GPR.R]                     ; MOVE SELECTED GPR TO MDR FOR TYPEING.
                                ;6807
                                ;6808  906:     ;*********FORCE ADDRESS*********;
                                ;6809  CN.E.GPR.IPR:
                                ;6810          ;---------------------------------;
U 0906, 04A6,95BE,4032,4047,008F,6 ;6811          M[TEMP9]_R[TEMP9],STEPC_2        ; IGNORE ANY SIZE SWITCH TYPED.
                                ;6812
                                ;6813          ;---------------------------------;
U 08F6, 0C9C,0735,C022,4047,008F,F ;6814          R[TEMP9]_8.DEC STEPC             ; LOAD # OF DIGITS TO TYPE FOR RESULT
                                ;6815
                                ;6816          ;---------------------------------;
                                ;6817          R[TEMP10]_M[TEMP7]+CONX(1),      ; ADD 1 FOR NEW DEFAULT ADDRESS.
U 08FF, 0484,7711,0002,8047,008F,8 ;6818          NEXT/CN.TYPE.RESULT              ;
                                ;6819
                                ;6820  CN.E.IPR:
                                ;6821          ;---------------------------------;
U 0903, 0080,05BF,4033,0007,0090,4 ;6822          PSL_R[TEMP12]                    ; RESTORE PSL INCASE EXAMINING IPL
                                ;6823
                                ;6824          ;---------------------------------;
U 0904, 0986,6C37,0037,F847,0090,D ;6825          M[TEMP6]_ZLIT0[0FF]              ; CLOBBER TO AVOID TYPEOUT IF ERROR.
                                ;6826
                                ;6827          ;---------------------------------;
U 090D, 0C84,1592,4031,C047,0091,6 ;6828          R[TEMP7]_M[TEMP1]                ; SAVE IPR ADDRESS
                                ;6829
                                ;6830          ;---------------------------------;
                                ;6831          M[TEMP10]_ZLIT0[49],             ; LOAD 'I' FOR CN.TYPE.RESULT
U 0916, 0186,AC37,0032,4847,0039,8 ;6832          NEXT/MP.MFPR                     ; GO FETCH THE IPR
                                ;6833
                                ;6834  CN.E.PSL:
                                ;6835          ;---------------------------------;
U 091A, 0B80,067E,F807,F847,0098,4 ;6836          LONLIT_[3020FF00]                ; LOAD MASK FOR UNDEFINED BITS.
                                ;6837
                                ;6838  =0      ;0---------------------------------;
                                ;6839          MDR_M[TEMP1],ANDNOT,R[LONLIT],   ; MOVE PSL INTO MDR FOR TYPEOUT
U 0984, 0080,1003,803D,4467,049F,4 ;6840          PUSH,NEXT/CN.TYPE.CHAR           ; TYPE <CR><LF>
                                ;6841
                                ;6842          ;1---------------------------------;
U 0985, 0484,0735,C022,4047,008F,D ;6843          R[TEMP9]_8,NEXT/CN.TYPE.DATA     ; LOAD NUMBER OF DIGITS FOR TYPEOUT.
```

```
                                ;6844  .TOC "  Console Command Parser          :       DEPOSIT ROUTINE"
                                ;6845
                                ;6846  ;*****************************************************************************
                                ;6847  ;       Entry Point     CN.DO.DEPOSIT
                                ;6848  ;
                                ;6849  ;       This section processes all deposits.  If no data has been entered it
                                ;6850  ;       is detected here and the bell is rung.  If a default address is to
                                ;6851  ;       be used it is detected here.  MP.MTPR is called to deposit into IPR's.
                                ;6852  ;       The new default address is calculated and the size switch updated.
                                ;6853  ;*****************************************************************************
                                ;6854  =0
                                ;6855  CN.DO.DEPOSIT:
                                ;6856          ;0------------------------------;
                                ;6857          M[TEMP5]_ZLIT0[7],              ; NO DATA TO DEPOSIT YET
U 0986, 0D86,5C37,0030,3847,0093,4 ;6858       NEXT/CN.PARSE.COMMAND           ; RING BELL
                                ;6859
                                ;6860          ;1------------------------------;
U 0987, 0480,0592,4A70,0047,0098,8 ;6861       WB_M[TEMP0],WX.NE.0?            ; CHECK IF A NEW ADDRESS WAS SPECIFIED
                                ;6862
                                ;6863  =0      ;0------------------------------; NO ADDRESS WAS SPECIFIED
U 0988, 0886,15BE,4032,8047,0098,9 ;6864       M[TEMP1]_R[TEMP10]              ; USE ADDRESS FROM PREVIOUS COMMAND
                                ;6865
                                ;6866          ;1------------------------------;
                                ;6867          R[TEMP8]_M[TEMP8],CLEAR FLAG3,  ; CLEAR FLAG3 TO AVOID APT COFLICT
U 0989, 0C1C,8592,4272,0047,0093,C ;6868       WB<1-0>?                        ; UPDATE SAVED SPACE SWITCH AND BRANCH.
                                ;6869
                                ;6870  =00     ;00------------------------------;
                                ;6871          VA_R[TEMP7]_M[TEMP1],           ; LOAD ADDRESS TO EXAMINE AND SAVE IT
U 093C, 0C84,1592,4031,C4A7,0093,7 ;6872       NEXT/CN.DEPOSIT.PHY             ;
                                ;6873
                                ;6874          ;01------------------------------;
                                ;6875          VA_R[TEMP7]_M[TEMP1],           ; LOAD ADDRESS TO EXAMINE AND SAVE IT
U 093D, 0C84,1592,4031,C4A7,0093,7 ;6876       NEXT/CN.DEPOSIT.PHY             ;
                                ;6877
                                ;6878          ;10------------------------------;
                                ;6879          RNUM_R[TEMP1],                  ; LOAD ADDRESS TO DEPOSIT IN GPR
U 093E, 0081,D5BE,4030,4047,0094,4 ;6880       NEXT/CN.DEPOSIT.GPR             ;
                                ;6881
                                ;6882          ;11------------------------------; DEPOSIT INTO IPR
U 093F, 0CA0,1002,403D,8467,0092,8 ;6883       MDR_M[TEMP1],STEPC_2            ; MOVE IPR # TO MDR FOR MP.MTPR
                                ;6884
                                ;6885          ;------------------------------;
U 092B, 0098,05BF,4033,0007,0092,F ;6886       PSL_R[TEMP12],DEC STEPC         ; RESTORE PSL INCASE DEPOSIT TO IPL.
                                ;6887
                                ;6888          ;------------------------------; STEPC_0 FOR TBIA
U 092F, 049C,1E50,0032,8047,0093,3 ;6889       R[TEMP10]_M[TEMP1]+1,DEC STEPC  ; UPDATE DEFAULT ADDRESS
                                ;6890
                                ;6891          ;------------------------------;
U 0933, 0480,3592,5030,0047,0039,0 ;6892       Q_M[TEMP3],NEXT/MP.MTPR         ;LOAD SOURCE DATA FOR MP.MTPR
```

```
                                        ;6893  CN.DEPOSIT.PHY:
                                        ;6894          ;-------------------------------;
                                        ;6895          R[TEMP9]_M[TEMP9],              ;
U 0937, 0084,9592,4232,4047,008D,0      ;6896          WB<5-0>?,NEXT/CN.DEPOSIT.SIZE   ; UPDATE SAVED SIZE SWITCH AND BRANCH.
                                        ;6897  =0000
                                        ;6898  CN.DEPOSIT.SIZE:
                                        ;6899  =0010   ;0010-------------------------;
U 08D2, 0080,3592,4000,05D8,008D,3      ;6900          WRITE M[TEMP3],SIZE[BYTE]       ; DO THE DEPOSIT
                                        ;6901
                                        ;6902          ;0011-------------------------;
                                        ;6903          R[TEMP10]_M[TEMP7]+CONX(1),     ; UPDATE NEW DEFAULT ADDRESS.
U 08D3, 0C84,7711,0002,8047,008A,F      ;6904          NEXT/CN.PROMPT                  ;
                                        ;6905
                                        ;6906          ;0100-------------------------;
U 08D4, 0480,3592,4010,05D8,008D,5      ;6907          WRITE M[TEMP3],SIZE[WORD]       ; DO THE DEPOSIT
                                        ;6908
                                        ;6909          ;0101-------------------------;
                                        ;6910          R[TEMP10]_M[TEMP7]+CONX(2),     ; UPDATE NEW DEFAULT ADDRESS.
U 08D5, 0884,7711,0012,8047,008A,F      ;6911          NEXT/CN.PROMPT                  ;
                                        ;6912
                                        ;6913  =1000   ;1000-------------------------;
U 08D8, 0C80,3592,4020,05D8,0094,3      ;6914          WRITE M[TEMP3],SIZE[LONG]       ; DO THE DEPOSIT
                                        ;6915  =
                                        ;6916          ;-------------------------------;
                                        ;6917          R[TEMP10]_M[TEMP7]+CONX(4),     ; UPDATE NEW DEFAULT ADDRESS.
U 0943, 0884,7711,0022,8047,008A,F      ;6918          NEXT/CN.PROMPT                  ;
                                        ;6919
                                        ;6920  CN.DEPOSIT.GPR:
                                        ;6921          ;-------------------------------;
U 0944, 0484,1711,0002,8047,0094,B      ;6922          R[TEMP10]_M[TEMP1]+CONX(1)      ;
                                        ;6923
                                        ;6924          ;-------------------------------;
                                        ;6925          R[GPR.R]_M[TEMP3],              ; DEPOSIT INTO GPR
U 094B, 0C84,3592,403C,0047,008A,F      ;6926          NEXT/CN.PROMPT                  ;
                                        ;6927
                                        ;6928  0BEC:   ;*********FORCE ADDRESS*********;
                                        ;6929  CN.DEPOSIT.PSL:
                                        ;6930          ;0-------------------------------;
                                        ;6931          M[TEMP5]_ZLITO[7],              ; NO DATA TO DEPOSIT YET
U 0BEC, 0D86,5C37,0030,3847,0093,4      ;6932          NEXT/CN.PARSE.COMMAND           ; RING THE BELL
                                        ;6933
                                        ;6934  0BED:   ;1-------------------------------; DEPOSIT IN SAVED COPY OF PSL
                                        ;6935          R[TEMP12]_M[TEMP3],             ;
U 0BED, 0C84,3592,4033,0047,008A,F      ;6936          NEXT/CN.PROMPT                  ;
```

```
                                ;6937  .TOC "  Console Command Parser          :       START, NEXT, CONTINUE COMMANDS"
                                ;6938
                                ;6939  ;*****************************************************************************
                                ;6940  ;       Entry Points    CN.DO.START, CN.CLEAR.HALT, CN.DO.CONTINUE
                                ;6941  ;
                                ;6942  ;       For START, this section loads PC and does INIT and then clears
                                ;6943  ;       HALT and HALT PENDING and does an IRD1.
                                ;6944  ;       For NEXT, (takes CN.DO.CONTINUE entry) HALT is cleared and
                                ;6945  ;       HALT PENDING is set.
                                ;6946  ;       For CONTINUE, HALT and HALT PENDING are both cleared.
                                ;6947  ;*****************************************************************************
                                ;6948
                                ;6949  CN.DO.CONTINUE:
                                ;6950          ;-------------------------------;
                                ;6951          PSL_R[TEMP12],                  ; RESTORE SAVED PSL
U 094F, 0880,05BF,4033,0007,0095,3 ;6952       NEXT/CN.RESTORE.PC              ;
                                ;6953  =00
                                ;6954  CN.DO.START:
                                ;6955          ;00-----------------------------; NO ADDRESS HAS BEEN SPECIFIED.
U 0940, 0086,35BE,4037,C047,0094,1 ;6956       M[TEMP3]_R[RTMPGPR]             ; USE CURRENT CONTENTS OF PC.
                                ;6957
                                ;6958          ;01-----------------------------;
                                ;6959          R[RTMPGPR]_M[TEMP3],SET FLAG2,  ; MOVE NEW PC TO RTMPGPR, SET FLAG TO
U 0941, 0054,3592,4037,C047,0487,9 ;6960       PUSH,NEXT/IN.INIT               ; TELL INIT THIS IS NOT POWER UP.
                                ;6961
                                ;6962  CN.CLEAR.HALT:
                                ;6963          ;10---------- ----------------;
                                ;6964          M[TEMP3]_7LIT16[40],            ; SETUP TO CLEAR HALT & HALT PENDING
U 0942, 0D16,3D37,0032,0047,0095,3 ;6965       CLEAR FLAG2                     ; CLEAR FLAG2 TO AVOID CONFLICT WITH APT
                                ;6966  =
                                ;6967  CN.RESTORE.PC:
                                ;6968          ;-------------------------------;
U 0953, 08A0,05BE,4037,C487,0098,A ;6969       PC_R[RTMPGPR],STEPC_2           ; LOAD NEW PC, STEPC NEEDS TO GET A 1.
                                ;6970
                                ;6971  =0       ;0-------------------------------;
                                ;6972          MEMSCAR_M[FPDOFFSET]_0,         ; PREPARE TO RESTORE MME, AND CLEAR
                                ;6973          DEC STEPC,                      ; FPDOFFSET TO INDICATE NOT CONSOLE MODE
U 098A, 009F,C5B7,0030,0687,049F,4 ;6974       PUSH,NEXT/CN.TYPE.CHAR          ; STEPC = 1 TYPE <CR><LF>.
                                ;6975
                                ;6976          ,0:-------------------------------;
                                ;6977          CRAR_ZLIT16[0C0],               ; LOAD ADDRESS OF HALT BIT.
U 098B, 0D80,0D37,0436,0247,0094,6 ;6978       FLAG0?,NEXT/CN.RESTORE.MME      ; CHECK IF INIT WAS DONE
                                ;6979  =00
                                ;6980  =01
                                ;6981  CN.IRD1:
                                ;6982          ;01-----------------------------;
U 0945, 0480,0036,4030,0047,0083,1 ;6983       NEXT/BO.IRD1                    ; LEAVE CONSOLE
                                ;6984
                                ;6985  CN.RESTORE.MME:
                                ;6986          ;10-----------------------------;
U 0946, 0880,05BE,4033,4607,0094,7 ;6987       MEMSCR_R[TEMP13]                ; INIT NOT DONE SO RESTORE MME
                                ;6988
                                ;6989          ;11-----------------------------;
                                ;6990          CONREGS_M[TEMP3],               ; CLEAR HALT & POSSIBLY SET HALT PENDING
U 0947, 0C80,3592,4CB0,02C7,0094,5 ;6991       FPS3?,NEXT/CN.IRD1              ; LOOP UNTIL ACLO GOES AWAY.
```

```
                                ;6992  .TOC "  Console Command Parser          :       BOOT COMMAND"
                                ;6993
                                ;6994  ;*************************************************************************
                                ;6995  ;
                                ;6996  ;       Resources       R1      MBA address for booting
                                ;6997  ;                       R2      UNIBUS I/O page address for booting
                                ;6998  ;
                                ;6999  ;       Subroutines     BO.BOOT
                                ;7000  ;                       BO.BOOT_SUB
                                ;7001  ;                       MV.TEST
                                ;7002  ;
                                ;7003  ;       This section processes a Boot command, if no device was specified
                                ;7004  ;       control is simply passed to BO.BOOT. If a device was specified
                                ;7005  ;       the device specification is parsed here to set up the MBA
                                ;7006  ;       and UNIBUS I/O page addresses in R1 and R2 respectively, and the PC
                                ;7007  ;       is loaded with the address of the appropriate boot rom.
                                ;7008  ;*************************************************************************
                                ;7009
                                ;7010  =0
                                ;7011  CN.DO.BOOT:
                                ;7012          ;0-------------------------------;
                                ;7013          MEMSCAR_ZLIT24[0],STEPC_6,      ; LOAD ADDRESS OF MME TO TURN IT OFF.
U 098C, 01A8,0CB7,0030,0687,0095,4 ;7014       NEXT/CN.BOOT.WITH.DEV           ;
                                ;7015
                                ;7016          ;1-------------------------------;
U 098D, 0884,3592,4035,4047,0080,6 ;7017       R[R5]_M[TEMP3],NEXT/BO.BOOT     ;
                                ;7018
                                ;7019  CN.BOOT.WITH.DEV:
                                ;7020          ;-------------------------------;
U 0954, 0898,05B7,0030,0607,0095,6 ;7021       MEMSCR_0,DEC STEPC              ; TURN OFF MME.
                                ;7022
                                ;7023          ;-------------------------------;
U 0956, 0780,07F8,6FDF,F847,0095,8 ;7024       LONLIT_[0F20400]                ; LOAD ADDRESS OF FIRST BOOT ROM.
                                ;7025
                                ;7026          ;-------------------------------; STEPC=4
U 0958, 0C98,05BE,403D,4487,0098,E ;7027       PC_R[LONLIT],DEC STEPC          ; USE XB FOR CHECKING FOR PROPER ROM.
                                ;7028  =0
                                ;7029  CN.BOOT.CHK.DEV:
                                ;7030          ;0-------------------------------;
                                ;7031          WB_ZEXT(XB)-(R[TEMP1].RR.16) PC_PC+2, ; CHECK IF DEVICE NAME MATCHES.
                                ;7032          SIGND CMP?,                     ;
U 098E, 0481,729C,0B50,6047,0888,6 439* ;7033  NEXT/CN.BOOT.CHK.DEV.10          ;
                                ;7034
                                ;7035          ;1-------------------------------;
                                ;7036          M[TEMP7]_ZLIT24[33],STEPC_2,    ;
U 098F, 05A6,7CB7,0031,9847,0096,C ;7037       NEXT/CN.ERROR                   ; NO MATCH FOUND FOR BOOT DEVICE.
```

```
                                ;7038  =10
                                ;7039  CN.BOOT.CHK.DEV.10:
                                ;7040          ;10--------------------------------;
                                ;7041          PC_M[PC]+ZLIT0[0FE],              ; INCREMENT TO START OF NEXT ROM
U 0886, 0D81,AC11,0337,F487,0098,E ;7042       DBZ STEPC?,NEXT/CN.BOOT.CHK.DEV   ;
                                ;7043
                                ;7044          ;11--------------------------------;
U 0887, 0C81,A000,003D,4487,0095,A ;7045       PC_M[PC]-R[LONLIT]                ; SAVE OFFSET FOR ROM.
                                ;7046
                                ;7047          ;--------------------------------  ;
U 095A, 0D80,0D37,2037,9847,0095,F ;7048       D_ZLIT16[0F3]                     ; SETUP BASE FOR LOADING R1 AND R2
                                ;7049
                                ;7050          ;--------------------------------;
U 095F, 0584,1D92,0037,F847,0096,0 ;7051       RTEMP7_M[TEMP1].AND.ZLIT8[0FF]    ; ISOLATE DEVICE TYPE (A, B, C, OR D)
                                ;7052
                                ;7053          ;--------------------------------;
                                ;7054          WB_M[TEMP7].XOR.ZLIT8[41],        ; CHECK IF DEVICE TYPE IS A
U 0960, 0D80,7D93,4A32,0847,0099,0 ;7055       WX.EQ.0?                          ;
                                ;7056
                                ;7057  =0      ;0--------------------------------;
                                ;7058          WB_M[TEMP7].XOR.ZLIT8[42],        ; CHECK IF DEVICE TYPE IS B
U 0990, 0580,7D93,4A32,1047,0099,2 ;7059       WX.EQ.0?,NEXT/CN.BOOT.DEV.B       ;
                                ;7060
                                ;7061          ;1--------------------------------;
U 0991, 0186,8DB0,0034,0047,0096,2 ;7062       M[TEMP8]_D-ZLIT8[80]              ; VALUE FOR R1
                                ;7063
                                ;7064          ;--------------------------------;
                                ;7065          M[TEMP9]_D+ZLIT12[0CE],           ; VALUE FOR R2
U 0962, 0586,9D71,0036,7047,00A0,0 ;7066       NEXT/CN.BOOT.DEV.TYPE.OK          ;
                                ;7067  =0
                                ;7068  CN.BOOT.DEV.B:
                                ;7069          ;0--------------------------------;
                                ;7070          WB_M[TEMP7].XOR.ZLIT8[43],        ; CHECK IF DEVICE   PE IS C
U 0992, 0180,7D93,4A32,1847,0099,4 ;7071       WX.EQ.0?,NEXT/CN.BOOT.DEV.C       ;
                                ;7072
                                ;7073          ;1--------------------------------;
U 0993, 0586,8DB0,0033,0047,0096,B ;7074       M[TEMP8]_D-ZLIT8[60]              ; VALUE FOR R1
                                ;7075
                                ;7076          ;--------------------------------;
                                ;7077          M[TEMP9]_D+ZLIT12[08E],           ; VALUE FOR R2
U 096B, 0186,9D71,0034,7047,00A0,0 ;7078       NEXT/CN.BOOT.DEV.TYPE.OK          ;
                                ;7079  =0
                                ;7080  CN.BOOT.DEV.C:
                                ;7081          ;0--------------------------------;
                                ;7082          WB_M[TEMP7].XOR.ZLIT8[44],        ; CHECK IF DEVICE TYPE IS D
U 0994, 0180,7D93,4A32,2047,0099,6 ;7083       WX.EQ.0?,NEXT/CN.BOOT.DEV.D       ;
                                ;7084
                                ;7085          ;1--------------------------------;
U 0995, 0186,8DB0,0032,0047,0097,0 ;7086       M[TEMP8]_D-ZLIT8[40]              ; VALUE FOR R1
                                ;7087
                                ;7088          ;--------------------------------;
                                ;7089          M[TEMP9]_D+ZLIT12[04E],           ; VALUE FOR R2
U 0970, 0186,9D71,0032,7047,00A0,0 ;7090       NEXT/CN.BOOT.DEV.TYPE.OK          ;
```

```
                                        ;7091  =0
                                        ;7092  CN.BOOT.DEV.D:
                                        ;7093          ;0-------------------------------;
                                        ;7094          M[TEMP7]_ZLIT24[34],STEPC_2,     ;
U 0996, 0DA6,7CB7,0031,A047,0096,C      ;7095          NEXT/CN.ERROR                    ; TYPE MUST BE A, B, C, OR D
                                        ;7096
                                        ;7097          ;1-------------------------------;
U 0997, 0986,8DB0,0031,0047,0097,2      ;7098          M[TEMP8]_D-ZLIT8[20]             ; VALUE FOR R1
                                        ;7099
                                        ;7100          ;-------------------------------;
                                        ;7101          M[TEMP9]_D+ZLIT12[00E],          ; VALUE FOR R2
U 0972, 0586,9D71,0030,7047,00A0,0      ;7102          NEXT/CN.BOOT.DEV.TYPE.OK         ;
                                        ;7103
                                        ;7104  CN.BOOT.DEV.TYPE.OK:
                                        ;7105          ;-------------------------------;
                                        ;7106          WB_M[TEMP1]-ZLIT0[41],           ; CHECK IF DEVICE NUMBER 0-9 OR A-F
U 0A00, 0180,1C10,0B42,0847,0889,D 413* ;7107          SIGND CMP DEF?,SIZE[BYTE]        ;
                                        ;7108
                                        ;7109  =01     ;01------------------------------;
U 089D, 0586,1C11,0030,4847,0089,F      ;7110          M[TEMP1]_MB+ZLIT0[9]             ; CORRECT LOW NIBBLE IF A-F
                                        ;7111
                                        ;7112          ;11------------------------------;
U 089F, 0884,13F7,0034,C047,00A0,1      ;7113          R[R3]_M[TEMP1].NIBBLE            ; LOAD DEVICE NUMBER
                                        ;7114
                                        ;7115          ;-------------------------------;
U 0A01, 0C85,A592,4037,C047,00A0,2      ;7116          R[RTMPGPR]_M[PC]                 ; SAVE OFFSET FOR NEW PC
                                        ;7117
                                        ;7118          ;-------------------------------;
U 0A02, 0084,8592,4034,4047,00A0,3      ;7119          R[R1]_M[TEMP8]                   ; LOAD MBA ADDRESS
                                        ;7120
                                        ;7121          ;-------------------------------;
U 0A03, 0C84,9592,4034,8047,0099,8      ;7122          R[R2]_M[TEMP9]                   ; LOAD UNIBUS I/O PAGE ADDRESS
                                        ;7123
                                        ;7124  =0      ;0------------------------------;
                                        ;7125          R[R5]_M[TEMP3],                  ; LOAD BOOT FLAGS
U 0998, 0084,3592,4035,4047,0482,B      ;7126          PUSH,NEXT/BO.BOOT_SUB            ;
                                        ;7127
                                        ;7128          ;1-------------------------------;
U 0999, 0885,A001,0037,C047,00A0,4      ;7129          R[RTMPGPR]_M[PC]+RB              ; LOAD NEW PC
                                        ;7130
                                        ;7131          ;-------------------------------;
U 0A04, 0081,6036,42B0,0047,0099,A      ;7132          WB_RNUM,WB<0>?                   ; CHECK IF MICRO VERIFY IS DISABLED.
                                        ;7133
                                        ;7134  =0      ;0------------------------------;
U 099A, 0C80,0036,4030,0047,0562,C      ;7135          PUSH,NEXT/MV.TEST                ; DO MICRO VERIFY.
                                        ;7136
                                        ;7137          ;1-------------------------------;
U 099B, 0C40,0036,4030,0047,0094,0      ;7138          SET FLAG0,NEXT/CN.DO.START       ;
```

```
                                ;7139  .TCC "  Console Command Parser          :       X(binary load/unload)  COMMAND"
                                ;7140
                                ;7141  ;*****************************************************************************
                                ;7142  ;
                                ;7143  ;       Resources       TEMP1           Memory address to read/write from/to
                                ;7144  ;                       TEMP3           Byte count
                                ;7145  ;                       TEMP5           One byte send/receive buffer
                                ;7146  ;                       TEMP6           Checksum of bytes sent
                                ;7147  ;                       TEMP7           Checksum of bytes read
                                ;7148  ;
                                ;7149  ;       For operation of the X command refer to the Midrange Systems
                                ;7150  ;       Console Language Specification under control of the VAX architecture
                                ;7151  ;       group.
                                ;7152  ;
                                ;7153  ;       Briefly, the command bytes including the checksum byte should add
                                ;7154  ;       to zero.  Likewise on a load/unload the bytes sent/received when
                                ;7155  ;       added together including the checksum byte should total zero.
                                ;7156  ;*****************************************************************************
                                ;7157  =0
                                ;7158  CN.DO.X:
                                ;7159          ;0-------------------------------;
                                ;7160          MEMSCR_0,                       ; CLEAR MME
                                ;7161          SET FLAG2,                      ; SET APT FLAG (RECEIVE BINARY MODE)
U 099C, 0850,05B7,0030,0607,049F,A ;7162       PUSH,NEXT/CN.GET.CHAR           ; GET CHECKSUM BYTE.
                                ;7163
                                ;7164          ;1-------------------------------;
                                ;7165          WB_M[TEMP7].AND.ZLIT0[OFF],     ;
U 099D, 0980,7C12,0A37,F847,0099,E ;7166       WX.EQ.0?                        ; CHECK IF CHECKSUM IS ZERO
                                ;7167
                                ;7168  =0      ;0-------------------------------;
                                ;7169          M[TEMP7]_ZLIT24[30],CLEAR FLAG2,; CHECKSUM ERROR
U 099E, 0D16,7CB7,0031,8047,0096,C ;7170       NEXT/CN.ERROR                   ;
                                ;7171
                                ;7172          ;1-------------------------------;
U 099F, 0D80,0C37,0030,1907,009A,0 ;7173       STEPC_ZLIT0[3]                  ; PREPARE TO SEND >>>
                                ;7174
                                ;7175  =0      ;0-------------------------------;
                                ;7176          M[TEMP5]_ZLIT0[3E],             ; SEND >>>
U 09A0, 0586,5C37,0031,F047,049F,4 ;7177       PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;7178
                                ;7179          ;1-------------------------------;
U 09A1, 0880,3592,46F0,0047,008C,9 ;7180       WB_M[TEMP3].WB<31-30>?          ; CHECK IF LOAD OR DUMP
                                ;7181
                                ;7182  =01     ;01------------------------------; X LOAD
                                ;7183          VA_M[TEMP1]-ZLIT0[1],           ; LOAD ADDRESS FOR LOADING
U 08C9, 0580,1C10,0030,0CA7,00A0,7 ;7184       NEXT/CN.X.LOAD                  ;
                                ;7185
                                ;7186          ;11------------------------------; X DUMP
U 08CB, 0986,3C93,8034,0047,00A0,5 ;7187       M[TEMP3]_MB.ANDNOT.ZLIT24[80]   ; CLEAR BIT 31 OF COUNT
                                ;7188
                                ;7189          ;-------------------------------;
U 0A05, 0C80,1002,403D,8487,00A0,6 ;7190       PC_M[TEMP1]                     ; LOAD ADDRESS TO BEGIN DUMP.
```

```
                                        ;7191           ;---------------------------------;
                                        ;7192           M[TEMP6]_R[ZERO],                 ; ZERO OUT TRANSMIT CHECKSUM
U 0A06, 0486,65BE,403D,8047,008F,2      ;7193           NEXT/CN.X.DUMP.10                 ;
                                        ;7194   =000
                                        ;7195   =001
                                        ;7196   CN.X.DUMP:
                                        ;7197           ;001-------------------------------;
                                        ;7198           R[TEMP5]_ZEXT(XB) PC_PC+1,        ; GET A BYTE FROM MEMORY TO SEND
U 08F1, 0C85,759E,4001,6047,049F,4      ;7199           PUSH,NEXT/CN.TYPE.CHAR            ; SEND IT.
                                        ;7200
                                        ;7201   CN.X.DUMP.10:
                                        ;7202           ;010-------------------------------;
                                        ;7203           M[TEMP3]_MB-1,WB<31-30>?,         ; DECREMENT COUNT AND CHECK IF NEGATIVE
U 08F2, 0C86,3080,06FD,8047,088F,1 344* ;7204           NEXT/CN.X.DUMP                    ;
                                        ;7205
                                        ;7206           ;011-------------------------------;
                                        ;7207           R[TEMP5]_-M[TEMP6],               ; SEND COMPLIMENT OF CHECKSUM
U 08F3, 0484,6593,0031,4047,049F,4      ;7208           PUSH,NEXT/CN.TYPE.CHAR            ;
                                        ;7209
                                        ;7210           ;100-------------------------------;
U 08F4, 0C10,0036,4030,0047,008A,F      ;7211           CLEAR FLAG2,NEXT/CN.PROMPT        ; CLEAR APT FLAG, GET NEXT COMMAND.
                                        ;7212   =
```

```
                                    ;7213  CN.X.LOAD:
                                    ;7214          ;--------------------------------;
U 0A07, 0886,75BE,403D,8047,00A0,8  ;7215          M[TEMP7]_R[ZERO]                ; CLEAR CHECKSUM
                                    ;7216
                                    ;7217  CN.X.LOAD.LOOP:
                                    ;7218          ;--------------------------------;
                                    ;7219          VA_M[VA]+CONX(1),               ; INCREMENT DESTINATION ADDRESS.
U 0A08, 0481,B711,0000,04A7,049F,B  ;7220          PUSH,NEXT/CN.GET.CHAR.NO.ECHO   ; GET NEXT BYTE
                                    ;7221
                                    ;7222          ;--------------------------------;
U 0A09, 0C86,3080,06FD,8047,088D,9 344* ;7223      M[TEMP3]_MB-1,WB<31-30>?         ; DECREMENT COUNT AND CHECK IF NEGATIVE
                                    ;7224
                                    ;7225  =01     ;01-------------------------------;
                                    ;7226          WRITE M[TEMP5],SIZE[BYTE],      ; LOAD 1 BYTE INTO MEMORY
U 08D9, 0880,5592,4000,05D8,00A0,8  ;7227          NEXT/CN.X.LOAD.LOOP             ; LOOP BACK FOR MORE
                                    ;7228
                                    ;7229          ;11-------------------------------;
                                    ;7230          WB_M[TEMP7].AND.ZLIT0[OFF],     ; CHECK LOW BYTE OF CHECKSUM FOR ZERO
U 08DB, 0980,7C12,0A37,F847,009A,2  ;7231          WX.EQ.0?                        ;
                                    ;7232
                                    ;7233  =0      ;0-------------------------------;
                                    ;7234          M[TEMP7]_ZLIT24[30],CLEAR FLAG2,; CHECKSUM ERROR
U 09A2, 0D16,7CB7,0031,8047,0096,C  ;7235          NEXT/CN.ERROR                   ;
                                    ;7236
                                    ;7237          ;1--------------------------------;
                                    ;7238          M[TEMP7]_R[ZERO],CLEAR FLAG2,   ; CLEAR COMMAND CHECKSUM AND APT FLAG
U 09A3, 0816,75BE,403D,8047,008A,F  ;7239          NEXT/CN.PROMPT                  ;
```

```
                                ;7240  .TOC "  Console Command Parser          : General Console Routines"
                                ;7241  .TOC "  Console Command Parser          :       CN.TYPE.RESULT"
                                ;7242
                                ;7243  ;*****************************************************************************
                                ;7244  ;       Entry points    CN.TYPE.RESULT
                                ;7245  ;                       CN.TYPE.DATA
                                ;7246  ;
                                ;7247  ;       Input           MDR             Number to be typed
                                ;7248  ;                       TEMP7           Address to be typed
                                ;7249  ;                       RTEMP9          Number of digits in result
                                ;7250  ;                       MTEMP10         Space indicator (P, V, I, G)
                                ;7251  ;
                                ;7252  ;       Jumping to CN.TYPE.RESULT will type the following:
                                ;7253  ;
                                ;7254  ;       <CR><LF>6 spaces<MTEMP10 byte 0>6 spaces<TEMP7>6 spaces<MDR>
                                ;7255  ;
                                ;7256  ;       Jumping to CN.TYPE.DATA will type the following:
                                ;7257  ;
                                ;7258  ;       6 spaces<MDR>
                                ;7259  ;*****************************************************************************
                                ;7260  =000
                                ;7261  CN.TYPE.RESULT:
                                ;7262          ;000-------------------------;
                                ;7263          M[TEMP5]_ZLIT0[0D],             ; TYPE <CR><LF>
U 08F8, 0586,5C37,0030,6847,049F,4 ;7264       PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;7265
                                ;7266          ;001-------------------------;
                                ;7267          M[TEMP5]_ZLIT0[20],STEPC_6,     ; TYPE 6 SPACES.
U 08F9, 09AE,5C37,0031,0047,049F,4 ;7268       PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;7269
                                ;7270          ;010-------------------------;
                                ;7271          R[TEMP5]_M[TEMP10],             ; TYPE SPACE INDICATOR (IE P,V,G,I)
U 08FA, 0484,A592,4031,4047,049F,4 ;7272       PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;7273
                                ;7274          ;011-------------------------;
                                ;7275          M[TEMP5]_ZLIT0[20],STEPC_6,     ; TYPE 6 SPACES.
U 08FB, 09AE,5C37,0031,0047,049F,4 ;7276       PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;7277
                                ;7278          ;100-------------------------; ADDRESS IS ALREADY IN TEMP7
                                ;7279          M[TEMP10]_ZLIT0[8],             ; SETUP # OF DIGITS FOR CN.TYPE.NUMBER
U 08FC, 0986,AC37,0030,4047,04A2,9 ;7280       PUSH,NEXT7CN.TYPE.NUMBER        ; TYPE NUMBER IN TEMP7.
```

```
                                ;7281  CN.TYPE.DATA:
                                ;7282          ;101-----------------------------;
                                ;7283          M[TEMP5]_ZLIT0[20],STEPC_6,     ; TYPE 6 SPACES.
 08FD, 09AE,5C37,0031,0047,049F,4 ;7284        PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;7285
                                ;7286          ;110-----------------------------; RTEMP9 IS # OF DIGITS IN RESULT.
 08FE, 0C86,A5BE,4032,4047,009A,4 ;7287        M[TEMP10]_R[TEMP9]              ; SETUP # OF DIGITS FOR CN.TYPE.NUMBER
                                ;7288  =
                                ;7289  =0      ;0-----------------------------;
                                ;7290          R[TEMP7]_M[MDR],                ; LOAD RESULT NUMBER TO BE TYPED.
 09A4, 0085,2592,4031,C047,04A2,9 ;7291        PUSH,NEXT/CN.TYPE.NUMBER        ;
                                ;7292
                                ;7293          ;1-----------------------------;
                                ;7294          R[TEMP9]_M[TEMP9],              ; UPDATE SAVED SIZE SWITCH FOR E/G.
 09A5, 0C84,9592,4032,4047,008A,F ;7295        NEXT/CN.PROMPT                  ; GET NEW COMMAND.
```

```
                                        ;7296  .TOC "  Console Command Parser          :       CN.GET.SWITCH"
                                        ;7297
                                        ;7298  ;*************************************************************************
                                        ;7299  ;
                                        ;7300  ;       This routine processes all switches for all commands.
                                        ;7301  ;
                                        ;7302  ;*************************************************************************
                                        ;7303  =00
                                        ;7304  CN.GET.SWITCH:
                                        ;7305          ;00-------------------------------;
U 0948, 0480,0036,4030,0047,049E,A      ;7306          PUSH,NEXT/CN.GET.NEXT           ; GET THE NEXT CHAR FROM THE CONSOLE.
                                        ;7307
                                        ;7308          ;01-------------------------------;
                                        ;7309          M[TEMP5]_ZLIT0[2F],             ; SETUP TO ECHO RUBOUT OF '/'.
U 0949, 0986,5C37,0081,7847,0000,1      ;7310          RETURN [+1]                     ;
                                        ;7311
                                        ;7312          ;10-------------------------------;
                                        ;7313          WB_M[TEMP5].XOR.ZLIT0[50],      ; CHECK IF /P SWITCH
U 094A, 0D80,5C13,4A72,8047,009A,6      ;7314          WX.NE.0?                        ;
                                        ;7315  =
                                        ;7316  =0      ;0-------------------------------;
                                        ;7317          WB_M[TEMP6].AND.CONX(1),        ; CHECK IF SPACE SWITCH HAS ALREADY BEEN
U 09A6, 0C80,6712,0A00,0047,009B,6      ;7318          WX.EQ.0?,NEXT/CN.P.SWITCH       ; SPECIFIED.
                                        ;7319
                                        ;7320          ;1 -------------------------------;
                                        ;7321          WB_M[TEMP5].XOR.ZLIT0[56],      ; CHECK IF /V SWITCH
U 09A7, 0980,5C13,4A72,B047,009A,8      ;7322          WX.NE.0?                        ;
                                        ;7323
                                        ;7324  =0      ;0-------------------------------;
                                        ;7325          WB_M[TEMP6].AND.CONX(1),        ; CHECK IF SPACE SWITCH HAS ALREADY BEEN
U 09A8, 0C80,6712,0A00,0047,009B,A      ;7326          WX.EQ.0?,NEXT/CN.V.SWITCH       ; SPECIFIED.
                                        ;7327
                                        ;7328          ;1-------------------------------;
                                        ;7329          WB_M[TEMP5].XOR.ZLIT0[47],      ; CHECK IF /G SWITCH
U 09A9, 0180,5C13,4A72,3847,009A,A      ;7330          WX.NE.0?                        ;
                                        ;7331
                                        ;7332  =0      ;0-------------------------------;
                                        ;7333          WB_M[TEMP6].AND.CONX(1),        ; CHECK IF SPACE SWITCH HAS ALREADY BEEN
U 09AA, 0480,6712,0A00,0047,009B,E      ;7334          WX.EQ.0?,NEXT/CN.G.SWITCH       ; SPECIFIED.
                                        ;7335
                                        ;7336          ;1-------------------------------;
                                        ;7337          WB_M[TEMP5].XOR.ZLIT0[49],      ; CHECK IF /I SWITCH
U 09AB, 0980,5C13,4A72,4847,009A,C      ;7338          WX.NE.0?                        ;
                                        ;7339
                                        ;7340  =0      ;0-------------------------------;
                                        ;7341          WB_M[TEMP6].AND.CONX(1),        ; CHECK IF SPACE SWITCH HAS ALREADY BEEN
U 09AC, 0C80,6712,0A00,0047,009C,2      ;7342          WX.EQ.0?,NEXT/CN.I.SWITCH       ; SPECIFIED.
```

```
                                ;7343           ;1------------------------------;
                                ;7344           WB_M[TEMP5].XOR.ZLIT0[42],      ; CHECK IF /B SWITCH
U C9AD, 0580,5C13,4A72,1047,009A,E ;7345        WX.NE.0?                        ;
                                ;7346
                                ;7347   =0      ;0------------------------------;
                                ;7348           WB_M[TEMP6].AND.CONX(2),        ; CHECK IF SIZE SWITCH HAS ALREADY BEEN
U 09AE, 0080,6712,0A10,0047,009C,6 ;7349        WX.EQ.0?,NEXT/CN.B.SWITCH       ; SPECIFIED.
                                ;7350
                                ;7351           ;1------------------------------;
                                ;7352           WB_M[TEMP5].XOR.ZLIT0[57],      ; CHECK IF /W SWITCH
U 09AF, 0D80,5C13,4A72,B847,009B,0 ;7353        WX.NE.0?                        ;
                                ;7354
                                ;7355   =0      ;0------------------------------;
                                ;7356           WB_M[TEMP6].AND.CONX(2),        ; CHECK IF SIZE SWITCH HAS ALREADY BEEN
U 09B0, 0080,6712,0A10,0047,009C,A ;7357        WX.EQ.0?,NEXT/CN.W.SWITCH       ; SPECIFIED.
                                ;7358
                                ;7359           ;1------------------------------;
                                ;7360           WB_M[TEMP5].XOR.ZLIT0[4C],      ; CHECK IF /L SWITCH
U 09B1, 0580,5C13,4A72,6047,009B,2 ;7361        WX.NE.0?                        ;
                                ;7362
                                ;7363   =0      ;0------------------------------;
                                ;7364           WB_M[TEMP6].AND.CONX(2),        ; CHECK IF SIZE SWITCH HAS ALREADY BEEN
U 09B2, 0880,6712,0A10,0047,009C,E ;7365        WX.EQ.0?,NEXT/CN.L.SWITCH       ; SPECIFIED.
                                ;7366
                                ;7367           ;1------------------------------;
                                ;7368           WB_M[TEMP5].XOR.ZLIT0[58],      ; CHECK IF /X SWITCH
U 09B3, 0980,5C13,4A72,C047,009B,4 ;7369        WX.NE.0?                        ;
                                ;7370
                                ;7371   =0      ;0------------------------------;
                                ;7372           WB_M[TEMP6].AND.ZLIT8[2],       ; CHECK IF /X SWITCH ALLOWED.
U 09B4, 0D80,6D92,0A30,1047,009D,2 ;7373        WX.EQ.0?,NEXT/CN.X.SWITCH       ;
                                ;7374
                                ;7375           ;1------------------------------; ASSUME /number IF NO OTHER MATCH
                                ;7376           WB_M[TEMP6].AND.ZLIT0[100],     ; CHECK IF /number SWITCH ALLOWED.
                                ;7377           SET FLAG1,                      ; INDICATE FIRST DIGIT HAS BEEN READ
U 09B5, 0D48,6C12,0A38,0047,0090,8 ;7378        WX.EQ.0?,NEXT/CN.H.SWITCH       ;
```

```
                                        ;7379  =0
                                        ;7380  CN.P.SWITCH:
                                        ;7381          ;0--------------------------------;
                                        ;7382          M[TEMP5]_ZLIT0[7],               ; SPACE SWITCH HAS ALREADY BEEN SET.
U 09B6, 0586,5C37,0030,3847,0094,8      ;7383          NEXT/CN.GET.SWITCH               ; RING BELL AND LOOK FOR NEW CHAR
                                        ;7384
                                        ;7385          ;1--------------------------------;
U 09B7, 0086,6712,4000,0047,00A0,A      ;7386          M[TEMP6]_MB.CR.CONX(1)           ; LOCK OUT OTHER SPACE SWITCHES.
                                        ;7387
                                        ;7388          ;--------------------------------;
U 0A0A, 0880,05B7,0030,0607,009B,8      ;7389          MEMSCR_0                         ; TURN OFF MME FOR PHYSICAL EXAMINE.
                                        ;7390
                                        ;7391  =0      ;0--------------------------------;
                                        ;7392          M[TEMP8]_R[ZERO],                ; SET SPACE SWITCH.
U 09B8, 0086,85BE,403D,8047,0493,4      ;7393          PUSH,NEXT/CN.PARSE.COMMAND       ; CONTINUE PARSING E COMMAND.
                                        ;7394
                                        ;7395          ;1--------------------------------; P HAS BEEN RUBBED OUT.
U 09B9, 0C86,85BE,4032,0047,00A0,B      ;7396          M[TEMP8]_R[TEMP8]                ; RESTORE SAVED SPACE SWITCH VALUE.
```

```
                                ;7397           ;--------------------------------;
U 0A0B, 0C80,83B7,0000,0607,00A0,C  ;7398       MEMSCR_M[TEMP8].RL.24           ; RESTORE MME TO SAVED VALUE.
                                ;7399
                                ;7400           ;--------------------------------;
U 0A0C, 0C86,6713,8000,0047,00A0,D  ;7401       M[TEMP6]_MB.ANDNOT.CONX(1)      ; ALLOW NEW SPACE SWITCH.
                                ;7402
                                ;7403           ;--------------------------------;
                                ;7404           M[TEMP5]_ZLIT0[50],             ; SETUP TO ECHO RUBOUT OF P.
U 0A0D, 0D86,5C37,0032,8047,0094,8  ;7405       NEXT/CN.GET.SWITCH              ;
                                ;7406   =0
                                ;7407   CN.V.SWITCH:
                                ;7408           ;0------------------------------;
                                ;7409           M[TEMP5]_ZLIT0[7],              ; SPACE SWITCH HAS ALREADY BEEN SET.
U 09BA, 0586,5C37,0030,3847,0094,8  ;7410       NEXT/CN.GET.SWITCH              ; RING BELL AND LOOK FOR NEW CHAR
                                ;7411
                                ;7412           ;1------------------------------;
U 09BB, 0886,6712,4000,0047,00A0,E  ;7413       M[TEMP6]_MB.OR.CONX(1)          ; LOCK OUT OTHER SPACE SWITCHES.
                                ;7414
                                ;7415           ;--------------------------------;
U 0A0E, 0480,0E77,0030,0607,009B,C  ;7416       MEMSCR_-1                       ; TURN ON MME FOR VIRTUAL EXAMINE.
                                ;7417
                                ;7418   =0      ;0------------------------------;
                                ;7419           M[TEMP8]_ZLIT0[1],              ; SET SPACE SWITCH.
U 09BC, 0D86,8C37,0030,0847,0493,4  ;7420       PUSH,NEXT/CN.PARSE.COMMAND      ; CONTINUE PARSING E COMMAND.
                                ;7421
                                ;7422           ;1------------------------------; V HAS BEEN RUBBED OUT.
U 09BD, 0486,85BE,4032,0047,00A0,F  ;7423       M[TEMP8]_R[TEMP8]               ; RESTORE SAVED SPACE SWITCH VALUE.
                                ;7424
                                ;7425           ;--------------------------------;
U 0A0F, 0480,83B7,0000,0607,00A1,0  ;7426       MEMSCR_M[TEMP8].RL.24           ; RESTORE MME TO SAVED VALUE.
                                ;7427
                                ;7428           ;--------------------------------;
U 0A10, 0486,6713,8000,0047,00A1,1  ;7429       M[TEMP6]_MB.ANDNOT.CONX(1)      ; ALLOW NEW SPACE SWITCH.
                                ;7430
                                ;7431           ;--------------------------------;
                                ;7432           M[TEMP5]_ZLIT0[56],             ; SETUP TO ECHO RUBOUT OF V.
U 0A11, 0186,5C37,0032,8047,0094,8  ;7433       NEXT/CN.GET.SWITCH              ;
                                ;7434   =0
                                ;7435   CN.G.SWITCH:
                                ;7436           ;0------------------------------;
                                ;7437           M[TEMP5]_ZLIT0[7],              ; SPACE SWITCH HAS ALREADY BEEN SET.
U 09BE, 0586,5C37,0030,3847,0094,8  ;7438       NEXT/CN.GET.SWITCH              ; RING BELL AND LOOK FOR NEW CHAR
                                ;7439
                                ;7440           ;1------------------------------;
U 09BF, 0086,6712,4000,0047,009C,0  ;7441       M[TEMP6]_MB.OR.CONX(1)          ; LOCK OUT OTHER SPACE SWITCHES.
                                ;7442
                                ;7443   =0      ;0------------------------------;
                                ;7444           M[TEMP8]_CONX(2),               ; SET SPACE SWITCH.
U 09C0, 0086,8737,0010,0047,0493,4  ;7445       PUSH,NEXT/CN.PARSE.COMMAND      ; CONTINUE PARSING E COMMAND.
```

```
                                ;7446           ;1------------------------------; G HAS BEEN RUBBED OUT.
U 09C1, 0486,85BE,4032,0047,00A1,2      ;7447           M[TEMP8]_R[TEMP8]               ; RESTORE SAVED SPACE SWITCH VALUE.
                                ;7448
                                ;7449           ;------------------------------;
U 0A12, 0C86,6713,8000,0047,00A1,3      ;7450           M[TEMP6]_MB.ANDNOT.CONX(1)      ; ALLOW NEW SPACE SWITCH.
                                ;7451
                                ;7452           ;------------------------------;
                                ;7453           M[TEMP5]_ZLIT0[47],             ; SETUP TO ECHO RUBOUT OF G.
U 0A13, 0186,5C37,0032,3847,0094,8      ;7454           NEXT/CN.GET.SWITCH              ;
                                ;7455   =0
                                ;7456   CN.I.SWITCH:
                                ;7457           ;0------------------------------;
                                ;7458           M[TEMP5]_ZLIT0[7],              ; SPACE SWITCH HAS ALREADY BEEN SET.
U 09C2, 0586,5C37,0030,3847,0094,8      ;7459           NEXT/CN.GET.SWITCH              ; RING BELL AND LOOK FOR NEW CHAR
                                ;7460
                                ;7461           ;1------------------------------;
U 09C3, 0886,6712,4000,0047,009C,4      ;7462           M[TEMP6]_MB.OR.CONX(1)          ; LOCK OUT OTHER SPACE SWITCHES.
                                ;7463
                                ;7464   =0      ;0------------------------------;
                                ;7465           M[TEMP8]_ZLIT0[3],              ; SET SPACE SWITCH.
U 09C4, 0986,8C37,0030,1847,0493,4      ;7466           PUSH,NEXT/CN.PARSE.COMMAND      ; CONTINUE PARSING E COMMAND.
                                ;7467
                                ;7468           ;1------------------------------; I HAS BEEN RUBBED OUT.
U 09C5, 0486,85BE,4032,0047,00A1,4      ;7469           M[TEMP8]_R[TEMP8]               ; RESTORE SAVED SPACE SWITCH VALUE.
                                ;7470
                                ;7471           ;------------------------------;
U 0A14, 0C86,6713,8000,0047,00A1,5      ;7472           M[TEMP6]_MB.ANDNOT.CONX(1)      ; ALLOW NEW SPACE SWITCH.
                                ;7473
                                ;7474           ;------------------------------;
                                ;7475           M[TEMP5]_ZLIT0[49],             ; SETUP TO ECHO RUBOUT OF I.
U 0A15, 0986,5C37,0032,4847,0094,8      ;7476           NEXT/CN.GET.SWITCH              ;
                                ;7477   =0
                                ;7478   CN.B.SWITCH:
                                ;7479           ;0------------------------------; SIZE SWITCH HAS ALREADY BEEN SET.
                                ;7480           WB_M[TEMP6].AND.ZLIT0[100],     ; CHECK IF /number SWITCH ALLOWED.
                                ;7481           SET FLAG1,                      ; INDICATE FIRST DIGIT HAS BEEN READ
U 09C6, 0D48,6C12,0A38,0047,0090,8      ;7482           WX.EQ.0?,NEXT/CN.H.SWITCH       ;
                                ;7483
                                ;7484           ;1------------------------------;
U 09C7, 0C86,6712,4010,0047,009C,8      ;7485           M[TEMP6]_MB.OR.CONX(2)          ; LOCK OUT OTHER SIZE SWITCHES.
                                ;7486
                                ;7487   =0      ;0------------------------------;
                                ;7488           M[TEMP9]_CONX(2),               ; SET SIZE SWITCH.
U 09C8, 0486,9737,0010,0047,0493,4      ;7489           PUSH,NEXT/CN.PARSE.COMMAND      ; CONTINUE PARSING COMMAND.
                                ;7490
                                ;7491           ;1------------------------------; B HAS BEEN RUBBED OUT.
U 09C9, 0C86,95BE,4032,4047,00A1,6      ;7492           M[TEMP9]_R[TEMP9]               ; RESTORE SAVED SIZE SWITCH VALUE.
                                ;7493
                                ;7494           ;------------------------------;
U 0A16, 0086,6713,8010,0047,00A1,7      ;7495           M[TEMP6]_MB.ANDNOT.CONX(2)      ; ALLOW NEW SIZE SWITCH.
                                ;7496
                                ;7497           ;------------------------------;
                                ;7498           M[TEMP5]_ZLIT0[42],             ; SETUP TO ECHO RUBOUT OF B.
U 0A17, 0D86,5C37,0032,1047,0094,8      ;7499           NEXT/CN.GET.SWITCH              ;
```

```
                                ;7500  =0
                                ;7501  CN.W.SWITCH:
                                ;7502          ;0------------------------------;
                                ;7503          M[TEMP5]_ZLIT0[7],              ; SIZE SWITCH HAS ALREADY BEEN SET.
U 09CA, 0586,5C37,0030,3847,0094,8 ;7504       NEXT/CN.GET.SWITCH              ; RING BELL AND LOOK FOR NEW CHAR
                                ;7505
                                ;7506          ;1------------------------------;
U 09CB, 0486,6712,4010,0047,009C,C ;7507       M[TEMP6]_MB.OR.CONX(2)          ; LOCK OUT OTHER SIZE SWITCHES.
                                ;7508
                                ;7509  =0      ;0------------------------------;
                                ;7510          M[TEMP9]_CONX(4),               ; SET SIZE SWITCH.
U 09CC, 0486,9737,0020,0047,0493,4 ;7511       PUSH,NEXT/CN.PARSE.COMMAND      ; CONTINUE PARSING COMMAND.
                                ;7512
                                ;7513          ;1------------------------------; W HAS BEEN RUBBED OUT.
U 09CD, 0486,95BE,4032,4047,00A1,8 ;7514       M[TEMP9]_R[TEMP9]               ; RESTORE SAVED SIZE SWITCH VALUE.
                                ;7515
                                ;7516          ;------------------------------;
U 0A18, 0886,6713,8010,0047,00A1,9 ;7517       M[TEMP6]_MB.ANDNOT.CONX(2)      ; ALLOW NEW SIZE SWITCH.
                                ;7518
                                ;7519          ;--- ----------------------------;
                                ;7520          M[TEMP5]_ZLIT0[57],             ; SETUP TO ECHO RUBOUT OF W.
U 0A19, 0586,5C37,0032,B847,0094,8 ;7521       NEXT/CN.GET.SWITCH              ;
                                ;7522  =0
                                ;7523  CN.L.SWITCH:
                                ;7524          ;0------------------------------;
                                ;7525          M[TEMP5]_ZLIT0[7],              ; SIZE SWITCH HAS ALREADY BEEN SET.
U 09CE, 0586,5C37,0030,3847,0094,8 ;7526       NEXT/CN.GET.SWITCH              ; RING BELL AND LOOK FOR NEW CHAR
                                ;7527
                                ;7528          ;1------------------------------;
U 09CF, 0C86,6712,4010,0047,009D,0 ;7529       M[TEMP6]_MB.OR.CONX(2)          ; LOCK OUT OTHER SIZE SWITCHES.
                                ;7530
                                ;7531  =0      ;0------------------------------;
                                ;7532          M[TEMP9]_ZLIT0[8],              ; SET SIZE SWITCH.
U 09D0, 0986,9C37,0030,4047,0493,4 ;7533       PUSH,NEXT/CN.PARSE.COMMAND      ; CONTINUE PARSING COMMAND.
                                ;7534
                                ;7535          ;1------------------------------; L HAS BEEN RUBBED OUT.
U 09D1, 0C86,95BE,4032,4047,00A1,A ;7536       M[TEMP9]_R[TEMP9]               ; RESTORE SAVED SIZE SWITCH VALUE.
                                ;7537
                                ;7538          ;------------------------------;
U 0A1A, 0086,6713,8010,0047,00A1,B ;7539       M[TEMP6]_MB.ANDNOT.CONX(2)      ; ALLOW NEW SIZE SWITCH.
                                ;7540
                                ;7541          ;------------------------------;
                                ;7542          M[TEMP5]_ZLIT0[4C],             ; SETUP TO ECHO RUBOUT OF L.
U 0A1B, 0586,5C37,0032,6047,0094,8 ;7543       NEXT/CN.GET.SWITCH              ;
```

```
                        ;7544  =000
                        ;7545  CN.H.SWITCH:
                        ;7546          ;000-----------------------------;
                        ;7547          M[TEMP5]_ZLIT0[7],CLEAR FLAG1,  ; H SWITCH NOT ALLOWED
U 0908, 050E,5C37,0030,3847,0094,8      ;7548          NEXT/CN.GET.SWITCH              ; RING BELL AND LOOK FOR NEW CHAR
                        ;7549
                        ;7550  CN.H.SWITCH.10:
                        ;7551          ;001-----------------------------;
                        ;7552          FLAG1?,SET FLAG0,               ; CHECK IF CHAR HAS ALREADY BEEN READ
U 0909, 0440,0036,45F0,C047,0491,0      ;7553          PUSH,NEXT/CN.GET.NUMBER         ; SET FLAG TO USE BUF
                        ;7554
                        ;7555          ;010-----------------------------;
                        ;7556          M[TEMP5]_ZLIT0[2F],             ; SETUP TO ECHO RUBOUT OF '/'.
U 090A, 0986,5C37,00B1,7847,0000,1      ;7557          RETURN [+1]                     ;
                        ;7558
                        ;7559          ;011-----------------------------;
                        ;7560          M[TEMP6]_MB.OR.ZLIT0[100],      ; ALLOW ONLY 1 H SWITCH
U 090B, 0186,6C12,4038,0047,0493.6      ;7561          PUSH,NEXT/CN.PARSE.RECURSE      ; CONTINUE PARSING COMMAND
                        ;7562
                        ;7563          ;100-----------------------------;
                        ;7564          M[TEMP6]_MB.ANDNOT.ZLIT0[100],  ; ALLOW NEW H SWITCH IF RUBBED OUT.
U 090C, 0586,6C13,8038,0047,049E,7      ;7565          PUSH,NEXT/CN.GET.NUM.RUBOUT.1   ; RUBOUT LAST DIGIT OF NUMBER
                        ;7566
                        ;7567  =110    ;110-----------------------------;
                        ;7568          M[TEMP5]_ZLIT0[2F],CLEAR FLAG1, ; SETUP TO ECHO RUBOUT OF '/'.
U 090E, 090E,5C37,00B1,7847,0000,1      ;7569          RETURN [+1]                     ;
                        ;7570
                        ;7571          ;111-----------------------------;
                        ;7572          CLEAR FLAG1,                    ; INDICATE NEXT CHARACTER HAS NOT BEEN
U 090F, 0408,0036,4030,0047,0090,9      ;7573          NEXT/CN.H.SWITCH.10             ; READ.
```

```
                                ;7574   =0
                                ;7575   CN.X.SWITCH:
                                ;7576           ;0------------------------------;
                                ;7577           M[TEMP5]_ZLIT0[7],              ; X SWITCH NOT ALLOWED.
U 09D2, 0586,5C37,0030,3847,0094,8  ;7578       NEXT/CN.GET.SWITCH              ; RING BELL AND LOOK FOR NEW CHAR
                                ;7579
                                ;7580           ;1------------------------------;
U 09D3, 0986,6D92,4030,1047,009D,4  ;7581       M[TEMP6]_MB.OR.ZLIT8[2]         ; ALLOW ONLY 1 X SWITCH.
                                ;7582
                                ;7583   =0      ;0------------------------------;
                                ;7584           RNUM_1,                         ; DISABLE MICRO VERIFY FOR BOOT.
U 09D4, 0481,DE74,0030,0047,0493,4  ;7585       PUSH,NEXT/CN.PARSE.COMMAND      ; CONTINUE PARSING COMMAND.
                                ;7586
                                ;7587           ;1------------------------------; X HAS BEEN RUBBED OUT.
U 09D5, 0881,D5B7,0030,0047,00A1,C  ;7588       RNUM_0                          ; ENABLE MICRO VERIFY FOR BOOT.
                                ;7589
                                ;7590           ;------------------------------;
U 0A1C, 0D86,6D93,8030,1047,00A1,D  ;7591       M[TEMP6]_MB.ANDNOT.ZLIT8[2]     ; ALLOW NEW X SWITCH.
                                ;7592
                                ;7593           ;------------------------------;
                                ;7594           M[TEMP5]_ZLIT0[58],             ; SETUP TO ECHO RUBOUT OF X.
U 0A1D, 0986,5C37,0032,C047,0094,8  ;7595       NEXT/CN.GET.SWITCH              ;
```

```
                                        ;7596  .TOC "  Console Command Parser          :       CN.GET.NUMBER"
                                        ;7597
                                        ;7598  ;*************************************************************************
                                        ;7599  ;
                                        ;7600  ;       This routine gets a number from the console and loads it either into
                                        ;7601  ;       TEMP1 or TEMP3.
                                        ;7602  ;       Special handling of *, +, and P for addresses is also done here.
                                        ;7603  ;*************************************************************************
                                        ;7604  =000
                                        ;7605  CN.GET.NUMBER:
                                        ;7606          ;000-------------------------------;
U 0910, 0480,0036,4030,0047,049E,A      ;7607          PUSH,NEXT/CN.GET.NEXT             ; ECHO CHAR IN TEMP5 AND GET NEXT CHAR.
                                        ;7608
                                        ;7609          ;001-------------------------------;
                                        ;7610          FLAG0?,                           ; RUBOUT FROM BUF0 OR BUF1?
U 0911, 0080,0036,4430,0047,049E,6      ;7611          PUSH,NEXT/CN.GET.NUM.RUBOUT       ;
                                        ;7612
                                        ;7613          ;010-------------------------------;
                                        ;7614          WB_M[TEMP5]-ZLIT0[30],            ; CHECK IF CHAR IS ASCII 0-9
U 0912, 0D80,5C10,06F1,8047,088E,D 377* ;7615          WB<31-30>?,NEXT/CN.GET.NUM.9      ;
                                        ;7616
                                        ;7617          ;011-------------------------------; RETURN FROM CN.GET.NUM.RUBOUT
U 0913, 0880,0036,40B0,0047,0000,1      ;7618          RETURN [+1]                       ; LAST DIGIT HAS BEEN RUBBED OUT.
                                        ;7619
                                        ;7620          ;100-------------------------------; RETURN FROM CN.GET.NUM.RUBOUT
U 0914, 0C80,0036,4030,0047,0091,0      ;7621          NEXT/CN.GET.NUMBER                ; SOME DIGITS REMAIN IN NUMBER.
                                        ;7622  =
                                        ;7623  =01
                                        ;7624  CN.GET.NUM.9:
                                        ;7625          ;01--------------------------------;
                                        ;7626          WB_M[TEMP5]-ZLIT0[3A],            ; CHECK IF CHAR IS ASCII 0-9
U 08ED, 0D80,5C10,06F1,D047,088F,5 377* ;7627          WB<31-30>?,NEXT/CN.GET.NUM.A      ;
                                        ;7628
                                        ;7629          ;11--------------------------------;
U 08EF, 0880,0036,4430,0047,009D,C      ;7630          FLAG0?,NEXT/CN.GET.NUMBER.END     ; IF EMPTY BUFFER, RING BELL
                                        ;7631  =01
                                        ;7632  CN.GET.NUM.A:
                                        ;7633          ;01--------------------------------;
                                        ;7634          WB_M[TEMP5]-ZLIT0[41],            ; CHECK IF CHAR IS ASCII A-F
U 08F5, 0580,5C10,06F2,0847,0890,5 377* ;7635          WB<31-30>?,NEXT/CN.GET.NUM.F      ;
```

```
                                ;7636  CN.GET.NIBBLE:
                                ;7637          ;11--------------------------------;
                                ;7638          R[TEMP4]_M[TEMP5].NIBBLE,       ; DIGIT IS ASCII 0-9 SO JUST USE LOW NIB
U 08F7, 0884,53F7,0431,0047,009D,6 ;7639       FLAG0?                          ; CHECK IF BUF0 OR BUF1.
                                ;7640  =0
                                ;7641  CN.GET.NUM.OVFLW:
                                ;7642          ;0--------------------------------;
                                ;7643          WB_M[TEMP0].XOR.ZLIT0[8],       ; CHECK IF BUF0 IS ALREADY FULL
U 09D6, 0180,0C13,4A70,4047,009D,A ;7644       WX.NE.0?,NEXT/CN.GET.NUM.BUF0   ;
                                ;7645
                                ;7646          ;1--------------------------------;
                                ;7647          WB_M[TEMP2].XOR.ZLIT0[8],       ; CHECK IF BUF1 IS ALREADY FULL
U 09D7, 0D80,2C13,4A70,4047,009D,8 ;7648       WX.NE.0?                        ;
                                ;7649
                                ;7650  =0      ;0--------------------------------;
                                ;7651          M[TEMP5]_ZLIT0[7],              ; BUF1 IS FULL, RING BELL
U 09D8, 0D86,5C37,0030,3847,0091,0 ;7652       NEXT/CN.GET.NUMBER              ;
                                ;7653
                                ;7654          ;1--------------------------------;
U 09D9, 0086,323E,4031,0047,00A1,E ;7655       M[TEMP3]_R[TEMP4].OR.((RB MB).RL.4) ; PUT DIGIT IN BUF1
                                ;7656
                                ;7657          ;--------------------------------;
                                ;7658          M[TEMP2]_MB+1,                  ; INCREMENT BUFCNT1
U 0A1E, 0C86,2081,003D,8047,0091,0 ;7659       NEXT/CN.GET.NUMBER              ; GO GET ANOTHER DIGIT.
                                ;7660  =0
                                ;7661  CN.GET.NUM.BUF0:
                                ;7662          ;0--------------------------------;
                                ;7663          M[TEMP5]_ZLIT0[7],              ; BUF0 IS FULL, RING BELL
U 09DA, 0D86,5C37,0030,3847,0091,0 ;7664       NEXT/CN.GET.NUMBER              ;
                                ;7665
                                ;7666          ;1--------------------------------;
U 09DB, 0C86,123E,4031,0047,00A1,F ;7667       M[TEMP1]_R[TEMP4].OR.((RB MB).RL.4) ; PUT DIGIT IN BUF0
                                ;7668
                                ;7669          ;--------------------------------;
                                ;7670          M[TEMP0]_MB+1,                  ; INCREMENT BUFCNT0
U 0A1F, 0886,0081,003D,8047,0091,0 ;7671       NEXT/CN.GET.NUMBER              ; GO GET ANOTHER DIGIT.
                                ;7672  =01
                                ;7673  CN.GET.NUM.F:
                                ;7674          ;01-------------------------------;
                                ;7675          WB_M[TEMP5]-ZLIT0[47],          ; IS CHAR ASCII A-F?
U 0905, 0180,5C10,06F2,3847,0891,5 377* ;7676  WB<31-30>?,NEXT/CN.GET.NUM.10     ;
                                ;7677
                                ;7678          ;11-------------------------------;
U 0907, 0880,0036,4430,0047,009D,C ;7679       FLAG0?,NEXT/CN.GET.NUMBER.END   ; IF EMPTY BUFFER, RING BELL
                                ;7680  =01
                                ;7681  CN.GET.NUM.10:
                                ;7682          ;01-------------------------------;
U 0915, 0880,0036,4430,0047,009D,C ;7683       FLAG0?,NEXT/CN.GET.NUMBER.END   ; IF EMPTY BUFFER, RING BELL
                                ;7684
                                ;7685          ;11-------------------------------;
U 0917, 0C84,53F7,0031,0047,00A2,0 ;7686       R[TEMP4]_M[TEMP5].NIBBLE        ; DIGIT IS ASCII 0-9 SO JUST USE LOW NIB
                                ;7687
                                ;7688          ;--------------------------------;
                                ;7689          M[TEMP4]_MB+ZLIT0[9],           ; ADJUST LOW NIB FOR ASCII A-F.
U 0A20, 0186,4C11,0430,4847,009D,6 ;7690       FLAG0?,NEXT/CN.GET.NUM.OVFLW    ; CHECK IF BUF0 OR BUF1.
```

```
                                        ;7691  =0
                                        ;7692  CN.GET.NUMBER.END:
                                        ;7693          ;0--------------------------------;
                                        ;7694          WB_M[TEMP0],                      ; CHECK IF BUFCNT0 .EQ. 0.
U 09DC, 0080,0592,4A30,0047,089E,0 322* ;7695          WX.EQ.0?,NEXT/CN.EMPTY.BUF        ;
                                        ;7696
                                        ;7697          ;1--------------------------------;
                                        ;7698          WB_M[TEMP2],                      ; CHECK IF BUFCNT1 .EQ. 0.
U 09DD, 0C80,2592,4A30,0047,089D,E 322* ;7699          WX.EQ.0?                          ;
                                        ;7700
                                        ;7701  =0      ;0--------------------------------;
U 09DE, 0880,0036,40B0,0047,0000,2      ;7702          RETURN [+2]                       ; BUFFER IS NOT EMPTY, NORMAL RETURN.
                                        ;7703
                                        ;7704          ;1--------------------------------;
                                        ;7705          M[TEMP5]_ZLIT0[7],                ; BUFFER IS EMPTY, NUMBER MUST HAVE
U 09DF, 0D86,5C37,0030,3847,0091,0      ;7706          NEXT/CN.GET.NUMBER                ; AT LEAST 1 DIGIT, RING BELL.
                                        ;7707  =0
                                        ;7708  CN.EMPTY.BUF:
                                        ;7709          ;0--------------------------------;
U 09E0, 0880,0036,40B0,0047,0000,2      ;7710          RETURN [+2]                       ; BUFFER IS NOT EMPTY, NORMAL RETURN.
                                        ;7711
                                        ;7712          ;1--------------------------------;
                                        ;7713          WB_M[TEMP5].XOR.ZLIT0[2A],        ; CHECK IF '*'
U 09E1, 0D80,5C13,4A31,5047,009E,2      ;7714          WX.EQ.0?                          ;
                                        ;7715
                                        ;7716  =0      ;0--------------------------------;
                                        ;7717          WB_M[TEMP5].XOR.ZLIT0[50],        ; CHECK IF 'P' FOR PSL
U 09E2, 0180,5C13,4A32,8047,0094,C      ;7718          WX.EQ.0?,NEXT/CN.GET.NUM.PSL      ;
                                        ;7719
                                        ;7720          ;1--------------------------------;
                                        ;7721          WB_M[TEMP8],                      ; BRANCH ON IF P,V OR G,I SPACE.
U 09E3, 0C80,8592,4270,0047,0091,9      ;7722          WB<1-0>?                          ;
                                        ;7723
                                        ;7724  =01     ;01-------------------------------; P OR V SPACE,
                                        ;7725          Q_R[TEMP9].SR.1,                  ; GET AMOUNT TO ADJUST ADDRESS BY.
U 0919, 0080,05BD,9032,4047,009E,4      ;7726          NEXT/CN.E.OR.D.STAR               ;
                                        ;7727
                                        ;7728          ;11-------------------------------; G OR I SPACE,
U 091B, 0D80,0C37,1030,0847,009E,4      ;7729          Q_ZLIT0[1]                        ; AMOUNT TO ADJUST ADDRESS BY IS 1.
                                        ;7730  =0
                                        ;7731  CN.E.OR.D.STAR:
                                        ;7732          ;0--------------------------------;
                                        ;7733          R[TEMP10]_RB-Q,                   ; BACKUP ADDRESS
U 09E4, 0084,0038,0032,8047,0493,4      ;7734          PUSH,NEXT/CN.PARSE.COMMAND        ;
                                        ;7735
                                        ;7736          ;1--------------------------------; RUBOUT
U 09E5, 0484,0039,0032,8047,00A2,1      ;7737          R[TEMP10]_RB+Q,NEXT/CN.RUB.STAR   ; RESTORE DEFAULT ADDRESS
```

```
                                ;7738  CN.RUB.STAR:
                                ;7739          ;--------------------------------;
                                ;7740          M[TEMP5]_ZLIT0[2A] Q_0,          ; PREPARE TO ECHO RUBOUT OF '*'
U 0A21, 0586,5C34,7031,5047,0091,0 ;7741       NEXT/CN.GET.NUMBER               ; CLEAR Q FOR DEFAULT ADDRESS IF ^U
                                ;7742  =00
                                ;7743  CN.GET.NUM.PSL:
                                ;7744          ;00--------------------------------; NOT PSL
                                ;7745          WB_M[TEMP5].XOR.ZLIT0[2B],       ; CHECK IF '+'
U 094C, 0980,5C13,4A31,5847,0095,0 ;7746       WX.EQ.0?,NEXT/CN.GET.NUM.PLUS    ;
                                ;7747
                                ;7748          ;01--------------------------------;
                                ;7749          M[TEMP6]_MB.OR.ZLIT0[80],        ; CHANGE E/D TO E/D PSL
U 094D, 0986,6C12,4034,0047,0493,4 ;7750       PUSH,NEXT/CN.PARSE.COMMAND       ; CONTINUE PARSING COMMAND.
                                ;7751
                                ;7752          ;10--------------------------------;
U 094E, 0D86,6C13,8034,0047,00A2,2 ;7753       M[TEMP6]_MB.ANDNOT.ZLIT0[80]     ; RESTORE TO NON PSL E/D
                                ;7754  =
                                ;7755          ;--------------------------------;
                                ;7756          M[TEMP5]_ZLIT0[50],              ; SETUP TO ECHO 'P'
U 0A22, 0586,5C37,0032,8047,0091,0 ;7757       NEXT/CN.GET.NUMBER               ;
                                ;7758  =00
                                ;7759  CN.GET.NUM.PLUS:
                                ;7760          ;00--------------------------------;
                                ;7761          M[TEMP5]_ZLIT0[7],               ; BUFFER IS EMPTY, NUMBER MUST HAVE
U 0950, 0D86,5C37,0030,3847,0091,0 ;7762       NEXT/CN.GET.NUMBER               ; AT LEAST 1 DIGIT, RING BELL.
                                ;7763
                                ;7764          ;C1--------------------------------;
U 0951, 0480,0036,4030,0047,0493,4 ;7765       PUSH,NEXT/CN.PARSE.COMMAND       ; CONTINUE PARSING COMMAND
                                ;7766
                                ;7767          ;10--------------------------------;
                                ;7768          M[TEMP5]_ZLIT0[2B],              ; PREPARE TO ECHO RUBOUT OF '+'
U 0952, 0586,5C37,0031,5847,0091,0 ;7769       NEXT/CN.GET.NUMBER               ;
                                ;7770  =
```

```
                                        ;7771  .TOC "  Console Command Parser          :       CN.GET.NUM.RUBOUT"
                                        ;7772
                                        ;7773  ;*****************************************************************************
                                        ;7774  ;
                                        ;7775  ;       This routine is used for rubbing out digits that have previously
                                        ;7776  ;       been loaded into TEMP1 or TEMP3.
                                        ;7777  ;*****************************************************************************
                                        ;7778  =0
                                        ;7779  CN.GET.NUM.RUBOUT:
                                        ;7780          ;0-------------------------------;
                                        ;7781          R[TEMP5]_M[TEMP1].NIBBLE,       ; GET DIGIT FOR RUBOUT ECHO.
U 09E6, 0C84,13F7,0031,4047,00A2,6      ;7782          NEXT/CN.GET.NUM.RO.5            ;
                                        ;7783
                                        ;7784  CN.GET.NUM.RUBOUT.1:
                                        ;7785          ;1-------------------------------;
U 09E7, 0884,33F7,0031,4047,00A2,3      ;7786          R[TEMP5]_M[TEMP3].NIBBLE        ; GET DIGIT FOR RUBOUT ECHO.
                                        ;7787
                                        ;7788          ;-------------------------------;
U 0A23, 0586,3C13,8030,7847,00A2,4      ;7789          M[TEMP3]_MB.ANDNOT.ZLITO[0F]    ;
                                        ;7790
                                        ;7791          ;-------------------------------;
U 0A24, 0C86,3277,0030,C047,00A2,5      ;7792          M[TEMP3]_(MB R[TEMP3]).RR.4     ; THROW AWAY LAST DIGIT.
                                        ;7793
                                        ;7794          ;-------------------------------;
                                        ;7795          M[TEMP2]_MB-CONX(1),            ; DECREMENT BUFCNT1
U 0A25, 0C86,2710,06C0,0047,0895,5 376* ;7796          WB<31-30>?                      ;
                                        ;7797
                                        ;7798  =01     ;01------------------------------;
                                        ;7799          WB_(M[TEMP5]+R[ZERO]).BCD,       ; SEPERATE 0-9 AND A-F
                                        ;7800          BCD CHECK?,                     ;
U 0955, 0880,5001,49AD,8047,089E,8 348* ;7801          NEXT/CN.GET.NUM.RO.BCD.CHK      ;
                                        ;7802
                                        ;7803          ;11------------------------------;
                                        ;7804          M[TEMP2]_R[ZERO],               ; ZERO BUFCNT1
U 0957, 0C86,25BE,40BD,8047,0000,2      ;7805          RETURN [+2]                     ; NO MORE TO RUBOUT.
                                        ;7806
                                        ;7807  CN.GET.NUM.RO.5:
                                        ;7808          ;-------------------------------;
U 0A26, 0186,1C13,8030,7847,00A2,7      ;7809          M[TEMP1]_MB.ANDNOT.ZLITO[0F]    ;
                                        ;7810
                                        ;7811          ;-------------------------------;
U 0A27, 0486,1277,0030,4047,00A2,8      ;7812          M[TEMP1]_(MB R[TEMP1]).RR.4     ; THROW AWAY LAST DIGIT.
```

```
                                        ;7813           ;-------------------------------;
                                        ;7814           M[TEMP0]_MB-CONX(1),            ; DECREMENT BUFCNTO
U 0A28, 0886,0710,06C0,0047,0895,9 376* ;7815           WB<31-30>?                      ;
                                        ;7816
                                        ;7817  =01      ;01-----------------------------;
                                        ;7818           WB_(M[TEMP5]+R[ZERO]).BCD,       ; SEPERATE 0-9 AND A-F
                                        ;7819           BCD CHECK?,                     ;
U 0959, 0880,5001,49AD,8047,089E,8 348* ;7820           NEXT/CN.GET.NUM.RO.BCD.CHK      ;
                                        ;7821
                                        ;7822           ;11-----------------------------;
                                        ;7823           M[TEMP0]_R[ZERO],               ; ZERO PUFCNTO
U 095B, 0886,05BE,40BD,8047,0000,2      ;7824           RETURN [+2]                     ; NO MORE TO RUBOUT.
                                        ;7825  =0
                                        ;7826  CN.GET.NUM.RO.BCD.CHK:
                                        ;7827           ;0-----------------------------;
                                        ;7828           M[TEMP5]_MB+ZLIT0[30],          ; ASCII 0-9
U 09E8, 0D86,5C11,00B1,8047,0000,3      ;7829           RETURN [+3]                     ;
                                        ;7830
                                        ;7831           ;1-----------------------------;
                                        ;7832           M[TEMP5]_MB+ZLIT0[37],          ; ASCII A-F
U 09E9, 0586,5C11,00B1,8847,0000,3      ;7833           RETURN [+3]                     ;
```

```
                                ;7834  .TOC "  Console Command Parser          :       CN.GET.NEXT (input character)"
                                ;7835
                                ;7836  ;*************************************************************************
                                ;7837  ;
                                ;7838  ;       This  outine gets one character at a time from the console and takes
                                ;7839  ;       care  f typing the '\'s that surround rubbed out characters.
                                ;7840  ;       If the last character typed was a rubout it does a RETURN+1,
                                ;7841  ;       if the last character typed was not a rubout it does a RETURN+2.
                                ;7842  ;*************************************************************************
                                ;7843  =0
                                ;7844  CN.GET.NEXT:
                                ;7845          ;0------------------------------;
U 09EA, 0C80,0036,4030,0047,049F,A  ;7846          PUSH,NEXT/CN.GET.CHAR          ; GET NEXT CHARACTER INTO TEMP5.
                                ;7847
                                ;7848          ;1------------------------------;
                                ;7849          WB_M[TEMP5].XOR.ZLIT0[5F],     ; CHECK FOR RUBOUT.
U 09EB, 0D80,5C13,4A32,F847,009E,C  ;7850          WX.EQ.0?                       ;
                                ;7851
                                ;7852  =0       ;0------------------------------; CHAR IS NOT RUBOUT
                                ;7853          D_M[TEMP5],MM.NOINT?,          ; SAVE CHAR INCASE NEED TO TYPE '\'
U 09EC, 0480,5002,647D,8047,0095,C  ;7854          NEXT/CN.GET.NEXT.NOT.RO        ; CHECK IF LAST CHAR WAS RUBOUT.
                                ;7855
                                ;7856          ;1------------------------------; CHAR IS RUBOUT
U 09ED, 0430,0036,4470,0047,009E,E  ;7857          MM.NOINT?                      ; CHECK IF LAST CHAR WAS RUBOUT.
                                ;7858
                                ;7859  =C       ;0------------------------------; LAST CHAR WAS NOT A RUBOUT.
                                ;7860          M[TEMP5]_ZLIT0[5C],SET MM.NOINT,; TYPE '\' TO INDICATE START OF RUBOUT.
U 09EF, 0966,5C37,0032,E047,049F,4  ;7861          PUSH,NEXT/CN.TYPE.CHAR         ;
                                ;7862
                                ;7863          ;1------------------------------; LAST CHAR WAS ALSO A RUBOUT.
U 09EF, 0880,0036,40B0,0047,0000,1  ;7864          RETURN [+1]                    ; TAKE RUBOUT RETURN.
                                ;7865  =C
                                ;7866  CN.GET.NEXT.NOT.RO:
                                ;7867          ;00-----------------------------;
U 095C, 0880,0036,40B0,0047,0000,2  ;7868          RETURN [+2]                    ; TAKE NORMAL RETURN.
                                ;7869
                                ;7870          ;01-----------------------------;
                                ;7871          M[TEMP5]_ZLIT0[5C],            ; TYPE '\' TO INDICATE END OF RUBOUT.
                                ;7872          CLEAR MM.NOINT,                ; CLEAR RUBOUT FLAG
U 095D, 0126,5C37,0032,E047,049F,4  ;7873          PUSH,NEXT/CN.TYPE.CHAR         ;
                                ;7874
                                ;7875          ;10-----------------------------;
                                ;7876          M[TEMP5]_D,                    ; RESTORE CHAR FROM DREG.
U 095E, 0886,55B2,40B0,0047,0000,2  ;7877          RETURN [+2]                    ; TAKE NORMAL RETURN.
                                ;7878  =
```

```
                                        ;7879  .TOC "  Console Command Parser          :       CN.TYPE.NUMBER"
                                        ;7880
                                        ;7881  ;*****************************************************************************
                                        ;7882  ;       Input           TEMP7           Number to be typed left justified
                                        ;7883  ;                       MTEMP10         Number of digits to type
                                        ;7884  ;
                                        ;7885  ;       Resources       TEMP5           Pass character to CN.TYPE.CHAR
                                        ;7886  ;
                                        ;7887  ;       Subroutines     CN.TYPE.CHAR
                                        ;7888  ;
                                        ;7889  ;       This routine will type 1-8 digits as specified by MTEMP10 starting
                                        ;7890  ;       with the leftmost nibble in TEMP7.
                                        ;7891  ;*****************************************************************************
                                        ;7892
                                        ;7893  CN.TYPE.NUMBER:
                                        ;7894          ;------------------------------;
U 0A29, 0086,7237,0031,C047,009F,0      ;7895          M[TEMP7]_(R[TEMP7] MB).RL.4     ; PUT MOST SIGFICANT DIGIT IN LOW NIBBLE
                                        ;7896  =0
                                        ;7897  CN.TYPE.NUMBER.LOOP:
                                        ;7898          ;0------------------------------;
                                        ;7899          R[TEMP5]_M[TEMP7].NIBBLE,       ; EXTRACT DIGIT TO BE TYPED.
U 09F0, 0C84,73F7,0031,4047,00AC,A      ;7900          NEXT/CN.TYPE.NUMBER.CONVERT     ;
                                        ;7901
                                        ;7902          ;1------------------------------;
U 09F1, 0880,0036,40B0,0047,0000,1      ;7903          RETURN [+1]                     ; NUMBER HAS BEEN TYPED, NORMAL RETURN.
                                        ;7904
                                        ;7905  CN.TYPE.NUMBER.CONVERT:
                                        ;7906          ;------------------------------;
                                        ;7907          WB_M[TEMP5]-ZLIT0[0A],          ; CHECK IF 0-9 OR A-F.
U 0A2A, 0D80,5C10,06F0,5047,0896,1 377* ;7908          WB<31-30>?                      ;
                                        ;7909
                                        ;7910  =01      ;01-----------------------------;
                                        ;7911          M[TEMP5]_MB+ZLIT0[37],          ; CONVERT A-F  TO ASCII A-F.
U 0961, 0986,5C11,0031,8847,009F,2      ;7912          NEXT/CN.TYPE.NUMBER.10          ;
                                        ;7913
                                        ;7914          ;11-----------------------------;
U 0963, 0186,5C11,0031,8047,009F,2      ;7915          M[TEMP5]_MB+ZLIT0[30]           ; CONVERT 0-9 TO ASCII 0-9.
                                        ;7916  =0
                                        ;7917  CN.TYPE.NUMBER.10:
                                        ;7918          ;0------------------------------;
                                        ;7919          M[TEMP7]_(R[TEMP7] MB).RL.4,    ; PUT NEXT DIGIT IN LOW NIBBLE.
U 09F2, 0086,7237,0031,C047,049F,4      ;7920          PUSH,NEXT/CN.TYPE.CHAR          ; TYPE THE DIGIT IN TEMP5.
                                        ;7921
                                        ;7922          ;1------------------------------;
                                        ;7923          M[TEMP10]_MB-CONX(1),           ; DECREMENT COUNT OF DIGITS IN NUMBER.
                                        ;7924          WX.EQ.0?,                       ; CHECK IF ANY MORE DIGITS.
U 09F3, 0886,A710,0A00,0047,089F,0 383* ;7925          NEXT/CN.TYPE.NUMBER.LOOP        ;
```

```
                                ;7926  .TOC "  Console Command Parser          :       CN.TYPE.CHAR"
                                ;7927  ;*************************************************************************
                                ;7928  ;       CN.TYPE.CHAR    TYPE DIGIT STEPC # OF TIMES
                                ;7929  ;               INPUT                   TEMP5           WHAT TO PRINT
                                ;7930  ;                                       STEPC           # OF TIMES TO PRINT
                                ;7931  ;                                                       MUST = 1 IF <CR> IS CHAR
                                ;7932  ;                                       FLAG2           SET APT AND
                                ;7933  ;                                                       <CR>.NE.<CR><LF>
                                ;7934  ;               OUTPUT                  STEPC           1(VALUE FOR FUTURE USE)
                                ;7935  ;               RESOURCES               ERRCOD          CHECK READY BIT
                                ;7936  ;                                       CRAR            NEED FOR STATUS,XMIT
                                ;7937  ;               NO SUBROUTINES
                                ;7938  ;*************************************************************************
                                ;7939
                                ;7940  9F4:    ;*********FORCE ADDRESS*********;
                                ;7941  CN.TYPE.CHAR:
                                ;7942          ;0-------------------------------;
                                ;7943          CRAR_ZLIT16[40],                ;CRAR GETS 1
U 09F4, 0580,0D37,0032,0247,009F,6  ;7944      NEXT/CN.TD.CHK.READY
                                ;7945
                                ;7946  9F5:    ;*********FORCE ADDRESS*********;
                                ;7947  CN.TYPE.CHAR.RET:
                                ;7948          ;1-------------------------------;
U 09F5, 0580,0C37,00B0,0907,0000,1  ;7949      STEPC_ZLIT0[1],RETURN [1]       ;STEPC GETS 1 RETURN
                                ;7950
                                ;7951  =0
                                ;7952  CN.TD.CHK.READY:
                                ;7953          ;0-------------------------------;
                                ;7954          M[ERRCOD]_CONREGS,              ;ERRCOD GETS STATUS+JUNK
U 09F6, 0886,B036,4030,03C7,00A2,8  ;7955      NEXT/CN.TD.IS.IT.READY          ;READS XCSR IN BIT 23
                                ;7956
                                ;7957          ;1-------------------------------;
U 09F7, 0980,0D37,0030,0247,00A2,C  ;7958      CRAR_ZLIT16[0],NEXT/CN.TD.XMIT  ;CRAR GETS 0 TRANSMIT
                                ;7959
                                ;7960  CN.TD.IS.IT.READY:
                                ;7961          ;-------------------------------;
                                ;7962          WB_M[ERRCOD].AND.ZLIT16[80],    ;XMIT READY?
U 0A2B, 0980,BD12,0A74,0047,009F,6  ;7963      WX.NE.0?,NEXT/CN.TD.CHK.READY   ;
                                ;7964
                                ;7965  CN.TD.XMIT:
                                ;7966          ;-------------------------------;
                                ;7967          CONREGS_M[TEMP5].RR.16,         ;XMIT DATA
U 0A2C, 0C80,53B7,0490,02C7,0088,8  ;7968      FLAG2?                          ;APT?
                                ;7969
                                ;7970  =0**
                                ;7971  CN.TD.XMIT.1:
                                ;7972          ;0**----------------------------; NOT APT, CHECK FOR <CR>
                                ;7973          WB_M[TEMP5]-ZLIT0[0D],          ;TYPING A <CR>?
                                ;7974          WX.EQ.0?,                       ;=0?
U 088B, 0580,5C10,0A30,6847,089F,8 384*  ;7975  NEXT/CN.TD.DEC.STEPC            ;
                                ;7976
                                ;7977          ;1**----------------------------; APT, DON'T CHANGE <CR> TO <CR><LF>
U 088F, 0C86,6001,0031,4047,009F,8  ;7978      M[TEMP6]_MB+R[TEMP5]            ; COMPUTE CHECKSUM FOR APT UNLOAD
```

```
                                        ;7979  =0
                                        ;7980  CN.TD.DEC,STEPC:
                                        ;7981          ;0-------------------------------;
U 09F8, 0C80,0036,4330,0047,009F,4      ;7982          DBZ STEPC?,NEXT/CN.TYPE.CHAR    ;NOT <CR> TYPE AGAIN OR FINISHED?
                                        ;7983
                                        ;7984          ;1-------------------------------;
                                        ;7985          M[TEMP5] ZLITO[0A],             ;YES <CR> TYPE LINE FEED
U 09F9, 0586,5C37,0030,5047,009F,4      ;7986          NEXT/CN.TYPE.CHAR               ;NEXT TIME THRU CHAR=A NOT D(<CR>)
```

```
                                ;7987  .TOC "  Console Command Parser          :       CN.GET.CHAR"
                                ;7988  ;*************************************************************************
                                ;7989  ;       CN.GET.CHAR     GET NEXT CHAR OFF FLY
                                ;7990  ;               INPUT                   TEMP5           OLD CHAR
                                ;7991  ;                                       FLAG2           APT INTERFACE
                                ;7992  ;                                       FLAG1           IN X CMD ONLY-BUT INTO
                                ;7993  ;                                                       GNC1(2 INST BELONG TO X)
                                ;7994  ;               OUTPUT                  TEMP5           NEW CHAR
                                ;7995  ;                                       RTMP4           TEMP6 XORED INTO RTMP4
                                ;7996  ;                                                       FOR APT INTERFACE
                                ;7997  ;                                       MTMP11          GETS RESTORED IF CNTRL U
                                ;7998  ;                                       RTMP9           GETS RESTORED IF CNTRL U
                                ;7999  ;                                       TEMP1           GETS RESTORED TO VA IF U
                                ;8000  ;               RESOURCES               STEPC           # OF X TO TYPE
                                ;8001  ;                                       CRAR            NEED FOR STATUS,RECEIVE
                                ;8002  ;               SUBROUTINES             CN.TYPE.CHAR
                                ;8003  ;                                       IE.ICR.SERV     FOR CLOCK SERVICE
                                ;8004  ;*************************************************************************
                                ;8005  =0
                                ;8006  CN.GET.CHAR:
                                ;8007          ;0-------------------------------;
                                ;8008          STEPC_ZLIT0[1],PUSH,            ;TYPE TEMP5 ONCE
U 09FA, 0980,0C37,0030,0907,049F,4  ;8009      NEXT/CN.TYPE.CHAR               ;
                                ;8010
                                ;8011  CN.GET.CHAR.NO.ECHO:
                                ;8012          ;1-------------------------------;
                                ;8013          CRAR_ZLIT16[80],                ;CRAR GETS 2
U 09FB, 0D80,0D37,0034,0247,0096,4  ;8014      NEXT/CN.IS.CHAR.READY           ;
                                ;8015  =00
                                ;8016  CN.IS.CHAR.READY:
                                ;8017          ;00------------------------------;
                                ;8018          M[TEMP5]_CONREGS,               ;TEMP5 GETS STATUS OF CRAR
                                ;8019          TIMER?,                         ;TIMER NEEDS SERVICE?
U 0964, 0C86,5036,4AF0,03C7,0096,6  ;8020      NEXT/CN.CHECK.DONE              ;
                                ;8021
                                ;8022  CN.GNC.CRAR.GETS.0:
                                ;8023          ;01------------------------------;
                                ;8024          CRAR_ZLIT16[0],                 ;CRAR GETS 0
                                ;8025          FPS3?,                          ;BIT 0 SET SAYS LOCKED
U 0965, 0D80,0D37,0CB0,0247,008C,6  ;8026      NEXT/CN.RECEIVE.CHAR            ;
                                ;8027
                                ;8028  CN.CHECK.DONE:
                                ;8029          ;10------------------------------;
                                ;8030          WB_M[TEMP5].AND.ZLIT16[80],     ;CHECK FOR DONE BIT
                                ;8031          WX.NE.0?,                       ;
U 0966, 0580,5D12,0A74,0047,0096,4  ;8032      NEXT/CN.IS.CHAR.READY           ;
                                ;8033
                                ;8034          ;11------------------------------;
                                ;8035          M[MM.TEMP0]_TCSR.IICR,          ;TO CHARLIE'S SUBR FOR CLOCK
U 0967, 0886,D036,4030,01E7,04FE,0  ;8036      NEXT/IE.ICR.SERV,PUSH           ;CHARLIE RETURNS-1
```

```
                                ;8037  =10
                                ;8038  CN.RECEIVE.CHAR:
                                ;8039          ;10-------------------------------;
                                ;8040          M[TEMP5]_CONREGS,               ;TEMP5 GETS CHARACTER
U 08C6, 0C86,5036,4030,03C7,00A2,D ;8041       NEXT/CN.ADJUST.CHAR             ;NOT LOCKED
                                ;8042
                                ;8043  =11
                                ;8044          ;11-------------------------------;
                                ;8045          M[TEMP5]_CONREGS,               ; READ CONREGS TO CLEAR DONE BIT
U 08C7, 0C86,5036,4030,03C7,0097,4 ;8046       NEXT/CN.TRY.CNTRL.P             ;LOCKED-RING THE BELL
                                ;8047
                                ;8048  CN.ADJUST.CHAR:
                                ;8049          ;-------------------------------;
                                ;8050          M[TEMP5]_MB.RR.16,              ;ADJUST CHAR <7:0>
                                ;8051          FLAG2?,                         ;APT INTERFACE?
U 0A2D, 0C86,53B7,0490,0047,0092,3 ;8052       NEXT/CN.STRIP.PAR               ;
                                ;8053
                                ;8054  =0**
                                ;8055  CN.STRIP.PAR:
                                ;8056          ;0**----------------------------;
                                ;8057          M[TEMP5]_MB.AND.ZLIT0[7F],      ;STRIP PARITY  NO APT
U 0923, 0186,5C12,0033,F847,00A2,E ;8058       NEXT/CN.IS.IT.LOWER.CASE        ;
                                ;8059
                                ;8060  CN.APT.INT:
                                ;8061          ;1**----------------------------;
                                ;8062          M[TEMP5]_MB.AND.ZLIT0[0FF],     ;GET BYTE   APT
U 0927, 0D86,5C12,0037,F847,0096,8 ;8063       NEXT/CN.CHKSUM.CHK              ;
                                ;8064
                                ;8065  CN.IS.IT.LOWER.CASE:
                                ;8066          ;-------------------------------;
                                ;8067          WB_M[TEMP5].AND.ZLIT0[40],      ;LOWER CASE IF WB NOT = 0
U 0A2E, 0D80,5C12,0A32,0047,009F,C ;8068       WX.EQ.0?                        ;
                                ;8069
                                ;8070  =0
                                ;8071          ;0-------------------------------;
U 09FC, 0986,5C12,0036,F847,009F,D ;8072       M[TEMP5]_MB.AND.ZLIT0[0DF]      ;LOWER CASE--STRIP OUT BIT SO UPPER CASE
                                ;8073
                                ;8074          ;1-------------------------------;
                                ;8075          WB_M[TEMP5]-ZLIT0[15],          ;DOES TEMP5=CTRLU?
U 09FD, 0580,5C10,0A30,A847,08EC,A 384* ;8076  WX.EQ.0?
                                ;8077
                                ;8078  0ECA:   ;*********FORCE ADDRESS*********;NOT 'U.
                                ;8079          WB_M[TEMP5]-ZLIT0[10],          ;IS IS ^P?
U 0ECA, 0980,5C10,0A30,8047,0896,8 384* ;8080  WX.EQ.0?,NEXT/CN.CHKSUM.CHK      ;
                                ;8081
                                ;8082  0ECB:   ;*********FORCE ADDRESS*********;
U 0ECB, 0480,0036,4030,0047,0096,9 ;8083       NEXT/CN.TYPE.CTRLU              ; ^U RUB LINE.
```

```
                                ;8084  =00
                                ;8085  CN.CHKSUM.CHK:
                                ;8086          ;00-------------------------------;
                                ;8087          M[TEMP7]_ZEXT(MB)+R[TEMP5],     ;TEMP5 ADDED INTO A BYTE OF TEMP5
                                ;80E8          SIZE[BYTE],                     ;ZERO EXTEND BY A BYTE
U 0968, 0C86,7015,0081,4047,0000,1              ;8089          RETURN [1]                      ;NO NOT =^U OR ^P.
                                ;8090
                                ;8091  CN.TYPE.CTRLU:
                                ;8092          ;01-------------------------------;
                                ;8093          M[TEMP5]_ZLIT0[5C],CLEAR FLAG1, ;TYPE / FOR CTRLU, CLEAR RUBOUT FLAG
U 0969, 010E,5C37,0032,E047,049F,4              ;8094          PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;8095
                                ;8096          ;10-------------------------------; CLEAR FLAG TO AVOID APT CONFLICT
U 096A, 041E,85BE,4032,0047,00A2,F              ;8097          M[TEMP8]_R[TEMP8],CLEAR FLAG3   ; RESTORE SAVED SPACE SWITCH
                                ;8098  =
                                ;8099          ;-------------------------------;
U 0A2F, 0484,0039,0032,8047,00A3,0              ;8100          R[TEMP10]_RB+Q                  ; RESTORE DEFAULT ADDRESS
                                ;8101
                                ;8102          ;-------------------------------;
                                ;8103          M[TEMP9]_R[TEMP9],              ; RESTORE SAVED SIZE SWITCH
U 0A30, 0C86,95BE,4032,4047,008A,F              ;8104          NEXT/CN.PROMPT                  ;
```

```
                                ;8105  .TOC "  Console Command Parser          :       CN.VIRT.TO.PHY"
                                ;8106
                                ;8107  ;*************************************************************************
                                ;8108  ;       INPUT                   MDR             CONTENTS OF VA
                                ;8109  ;                               VA              ADDRESS(VIRTUAL)
                                ;8110  ;       OUTPUT                  VA              "       "
                                ;8111  ;                               MDR             CONTENTS OF VA
                                ;8112  ;                               TEMP7           PHYSICAL ADDRESS
                                ;8113  ;       RESOURCES               TEMPO           GETS 0'MDR
                                ;8114  ;       SUBROUTINE              MM.GET.PTE
                                ;8115  ;*************************************************************************
                                ;8116
                                ;8117  =000
                                ;8118  CN.VIRT.TO.PHY:
                                ;8119          ;000-------------------------------;
                                ;8120          PUSH,                           ;RET+6,+7
                                ;8121          R[MM.TEMP3]_M[VA],              ;SAVE VA
                                ;8122          WB<31-30>?,                     ;BUT INTO SUBROUTINE
U 0918, 0485,B592,46F9,C047,055D,8  ;8123      NEXT/MM.GET.PTE                 ;GETS PAGE TABLE ENTRY
                                ;8124
                                ;8125  =100
                                ;8126          ;100-------------------------------;
                                ;8127          VA_M[TEMP7]_R[MM.TEMP3],        ;VA,TEMP7 GET VIRTUAL VA
U 091C, 0886,75BE,4039,C4A7,00A3,1  ;8128      NEXT/CN.RESTORE.TEMP7           ;GET VIRT ADDRESS INTO TEMP7
                                ;8129
                                ;8130          ;101-------------------------------;
                                ;8131          VA_M[TEMP7]_R[MM.TEMP3],        ;VA,TEMP7 GET VIRTUAL VA
U 091D, 0886,75BE,4039,C4A7,00A3,1  ;8132      NEXT/CN.RESTORE.TEMP7           ;GET VIRT ADDRESS INTO TEMP7
                                ;8133
                                ;8134          ;110-------------------------------;
                                ;8135          VA_M[TEMP7]_R[MM.TEMP3],        ;VA,TEMP7 GET VIRTUAL VA
U 091E, 0886,75BE,4039,C4A7,00A3,1  ;8136      NEXT/CN.RESTORE.TEMP7           ;GET VIRT ADDRESS INTO TEMP7
                                ;8137
                                ;8138          ;111-------------------------------;
U 091F, 0886,75BE,4039,C4A7,00A3,1  ;8139      VA_M[TEMP7]_R[MM.TEMP3]         ;VA,TEMP7 GET VIRTUAL VA
                                ;8140
                                ;8141  CN.RESTORE.TEMP7:
                                ;8142          ;-------------------------------;
U 0A31, 0D86,7C12,003F,F847,00A3,2  ;8143      M[TEMP7]_MB.AND.ZLITO[1FF]      ;TEMP7_9 BITS OF THE VIRTUAL ADDRESS
                                ;8144
                                ;8145          ;-------------------------------;
                                ;8146          R[TEMPO]_ZEXT(M[MDR]),          ;TEMPO_0(16 BITS)'MDR(16 BITS)
U 0A32, 0085,259E,4010,0047,00A3,3  ;8147      SIZE[WORD]                      ;WORD SIZE
                                ;8148
                                ;8149          ;-------------------------------;
U 0A33, 0884,047E,4031,C047,00A3,4  ;8150      R[TEMP7]_RB.OR.(M[TEMPO].RL.9)  ;GET UPPER BITS
                                ;8151
                                ;8152          ;-------------------------------;
                                ;8153          MDR_R[MM.TEMP2],                ;RESTORE MDR
U 0A34, 0480,05BE,40B9,8467,03FF,F  ;8154      RETURN [-1]                     ;RETURN-1
```

```
                                ;8155  .TOC "  Console Command Parser          :       CN.CLEAR.TEMPS"
                                ;8156
                                ;8157  ;*****************************************************************************
                                ;8158  ;*****************************************************************************
                                ;8159
                                ;8160  =01
                                ;8161  CN.CLEAR.TEMPS:
                                ;8162          ;01-------------------------------;
                                ;8163          R[TEMP.R] 0,                     ;
U 0971, 0484,05B7,003C,0047,00A3,5 ;8164          NEXT/CN.CLEAR.TEMPS.10           ;
                                ;8165
                                ;8166          ;11------------------------------;
                                ;8167          RNUM 0,                          ; ZERO INCASE BOOT WITHOUT /X
U 0973, 0C81,D5B7,00B0,0047,0000,1 ;8168          RETURN [+1]                      ;
                                ;8169
                                ;8170  0A35:    ;*********FORCE ADDRESS*********;
                                ;8171  CN.CLEAR.TEMPS.10:
                                ;8172          ;0-------------------------------;
                                ;8173          M[TEMP.R] 0,                     ;
U 0A35, 0C87,05B7,0030,0047,056B,6 ;8174          PUSH,NEXT7MV.SET.CRAR            ; LOAD CRAR FOR SETTING HALT LATER.
                                ;8175
                                ;8176  0A36:    ;1-------------------------------;
                                ;8177          RNUM D D-1,                      ;
U 0A36, 0C81,DQA0,26FD,8047,0897,1 344* ;8178          WB<3T-30>?,NEXT/CN.CLEAR.TEMPS   ;
```

```
                                ;8179  .TOC "  Console Command Parser          :       CN.ERROR"
                                ;8180
                                ;8181  ;*************************************************************************
                                ;8182  ;*************************************************************************
                                ;8183  96C:    ;*********FORCE ADDRESS*********;
                                ;8184  CN.ERROR:
                                ;8185          ;00-----------------------------;
                                ;8186          M[TEMP5]_ZLIT0[0D],             ; TYPE <CR><LF>
U 096C, 0586,5C37,0030,6847,049F,4 ;8187       PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;8188
                                ;8189  96D:    ;01-----------------------------;
                                ;8190          M[TEMP5]_ZLIT0[3F],             ; TYPE ?
U 096D, 0186,5C37,0031,F847,049F,4 ;8191       PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;8192
                                ;8193  96E:    ;10-----------------------------;
                                ;8194          M[TEMP10]_CONX(2),              ; LOAD NUMBER OF DIGITS TO TYPE.
U 096E, 0486,A737,0010,0047,04A2,9 ;8195       PUSH,NEXT/CN.TYPE.NUMBER        ; TYPE ERROR NUMBER
                                ;8196
                                ;8197  96F:    ;11-----------------------------;
                                ;8198          WB_M[TEMP6].AND.ZLIT0[0F0],     ; CHECK IF THIS WAS AN EXAMINE
U 096F, 0180,6C12,0A37,8047,009F,E ;8199       WX.EQ.0?                        ;
                                ;8200
                                ;8201  =0      ;0-----------------------------;
U 09FE, 0C10,0036,4030,0047,008A,F ;8202       CLEAR FLAG2,NEXT/CN.PROMPT      ; CLEAR APT FLAG JUST IN CASE.
                                ;8203
                                ;8204          ;1-----------------------------;
                                ;8205          M[TEMP10]_ZLIT0[3F],            ; TYPE ? AS SPACE CHARACTER TO INDICATE
U 09FF, 0186,AC37,0031,F847,008F,8 ;8206       NEXT/CN.TYPE.RESULT             ; ERROR. ON EXAMINE TYPE RESULT ANYWAY.
```

8206; This page intentionally left blank.

B	1	MICRO2	1M(01)	J	5	MICRO2	1M(01)	E	10	MICRO2	1M(01)
C	1	MICRO2	1M(01)	K	5	MICRO2	1M(01)	F	10	MICRO2	1M(01)
D	1	MICRO2	1M(01)	L	5	MICRO2	1M(01)	G	10	MICRO2	1M(01)
E	1	MICRO2	1M(01)	M	5	MICRO2	1M(01)	H	10	MICRO2	1M(01)
F	1	MICRO2	1M(01)	N	5	MICRO2	1M(01)	I	10	MICRO2	1M(01)
G	1	MICRO2	1M(01)	B	6	MICRO2	1M(01)	J	10	MICRO2	1M(01)
H	1	MICRO2	1M(01)	C	6	MICRO2	1M(01)	K	10	MICRO2	1M(01)
I	1	MICRO2	1M(01)	D	6	MICRO2	1M(01)	L	10	MICRO2	1M(01)
J	1	MICRO2	1M(01)	E	6	MICRO2	1M(01)	M	10	MICRO2	1M(01)
K	1	MICRO2	1M(01)	F	6	MICRO2	1M(01)	N	10	MICRO2	1M(01)
L	1	MICRO2	1M(01)	G	6	MICRO2	1M(01)	B	11	MICRO2	1M(01)
M	1	MICRO2	1M(01)	H	6	MICRO2	1M(01)	C	11	MICRO2	1M(01)
N	1	MICRO2	1M(01)	I	6	MICRO2	1M(01)	D	11	MICRO2	1M(01)
B	2	MICRO2	1M(01)	J	6	MICRO2	1M(01)	E	11	MICRO2	1M(01)
C	2	MICRO2	1M(01)	K	6	MICRO2	1M(01)	F	11	MICRO2	1M(01)
D	2	MICRO2	1M(01)	L	6	MICRO2	1M(01)	G	11	MICRO2	1M(01)
E	2	MICRO2	1M(01)	M	6	MICRO2	1M(01)	H	11	MICRO2	1M(01)
F	2	MICRO2	1M(01)	N	6	MICRO2	1M(01)	I	11	MICRO2	1M(01)
G	2	MICRO2	1M(01)	B	7	MICRO2	1M(01)	J	11	MICRO2	1M(01)
H	2	MICRO2	1M(01)	C	7	MICRO2	1M(01)	K	11	MICRO2	1M(01)
I	2	MICRO2	1M(01)	D	7	MICRO2	1M(01)	L	11	MICRO2	1M(01)
J	2	MICRO2	1M(01)	E	7	MICRO2	1M(01)	M	11	MICRO2	1M(01)
K	2	MICRO2	1M(01)	F	7	MICRO2	1M(01)	N	11	MICRO2	1M(01)
L	2	MICRO2	1M(01)	G	7	MICRO2	1M(01)	B	12	MICRO2	1M(01)
M	2	MICRO2	1M(01)	H	7	MICRO2	1M(01)	C	12	MICRO2	1M(01)
N	2	MICRO2	1M(01)	I	7	MICRO2	1M(01)	D	12	MICRO2	1M(01)
B	3	MICRO2	1M(01)	J	7	MICRO2	1M(01)	E	12	MICRO2	1M(01)
C	3	MICRO2	1M(01)	K	7	MICRO2	1M(01)	F	12	MICRO2	1M(01)
D	3	MICRO2	1M(01)	L	7	MICRO2	1M(01)	G	12	MICRO2	1M(01)
E	3	MICRO2	1M(01)	M	7	MICRO2	1M(01)	H	12	MICRO2	1M(01)
F	3	MICRO2	1M(01)	N	7	MICRO2	1M(01)	I	12	MICRO2	1M(01)
G	3	MICRO2	1M(01)	B	8	MICRO2	1M(01)	J	12	MICRO2	1M(01)
H	3	MICRO2	1M(01)	C	8	MICRO2	1M(01)	K	12	MICRO2	1M(01)
I	3	MICRO2	1M(01)	D	8	MICRO2	1M(01)	L	12	MICRO2	1M(01)
J	3	MICRO2	1M(01)	E	8	MICRO2	1M(01)	M	12	MICRO2	1M(01)
K	3	MICRO2	1M(01)	F	8	MICRO2	1M(01)	N	12	MICRO2	1M(01)
L	3	MICRO2	1M(01)	G	8	MICRO2	1M(01)	B	13	MICRO2	1M(01)
M	3	MICRO2	1M(01)	H	8	MICRO2	1M(01)	C	13	MICRO2	1M(01)
N	3	MICRO2	1M(01)	I	8	MICRO2	1M(01)	D	13	MICRO2	1M(01)
B	4	MICRO2	1M(01)	J	8	MICRO2	1M(01)	E	13	MICRO2	1M(01)
C	4	MICRO2	1M(01)	K	8	MICRO2	1M(01)	F	13	MICRO2	1M(01)
D	4	MICRO2	1M(01)	L	8	MICRO2	1M(01)	G	13	MICRO2	1M(01)
E	4	MICRO2	1M(01)	M	8	MICRO2	1M(01)	H	13	MICRO2	1M(01)
F	4	MICRO2	1M(01)	N	8	MICRO2	1M(01)	I	13	MICRO2	1M(01)
G	4	MICRO2	1M(01)	B	9	MICRO2	1M(01)	J	13	MICRO2	1M(01)
H	4	MICRO2	1M(01)	C	9	MICRO2	1M(01)	K	13	MICRO2	1M(01)
I	4	MICRO2	1M(01)	D	9	MICRO2	1M(01)	L	13	MICRO2	1M(01)
J	4	MICRO2	1M(01)	E	9	MICRO2	1M(01)	M	13	MICRO2	1M(01)
K	4	MICRO2	1M(01)	F	9	MICRO2	1M(01)	N	13	MICRO2	1M(01)
L	4	MICRO2	1M(01)	G	9	MICRO2	1M(01)	B	14	MICRO2	1M(01)
M	4	MICRO2	1M(01)	H	9	MICRO2	1M(01)	C	14	MICRO2	1M(01)
N	4	MICRO2	1M(01)	I	9	MICRO2	1M(01)	D	14	MICRO2	1M(01)
B	5	MICRO2	1M(01)	J	9	MICRO2	1M(01)	E	14	MICRO2	1M(01)
C	5	MICRO2	1M(01)	K	9	MICRO2	1M(01)	F	14	MICRO2	1M(01)
D	5	MICRO2	1M(01)	L	9	MICRO2	1M(01)	G	14	MICRO2	1M(01)
E	5	MICRO2	1M(01)	M	9	MICRO2	1M(01)	H	14	MICRO2	1M(01)
F	5	MICRO2	1M(01)	N	9	MICRO2	1M(01)	I	14	MICRO2	1M(01)
G	5	MICRO2	1M(01)	B	10	MICRO2	1M(01)	J	14	MICRO2	1M(01)
H	5	MICRO2	1M(01)	C	10	MICRO2	1M(01)	K	14	MICRO2	1M(01)
I	5	MICRO2	1M(01)	D	10	MICRO2	1M(01)	L	14	MICRO2	1M(01)

M 14 MICRO2 1M(01)
N 14 MICRO2 1M(01)
B 15 MICRO2 1M(01)
C 15 MICRO2 1M(01)
D 15 MICRO2 1M(01)
E 15 MICRO2 1M(01)
F 15 MICRO2 1M(01)
G 15 MICRO2 1M(01)
H 15 MICRO2 1M(01)
I 15 MICRO2 1M(01)
J 15 MICRO2 1M(01)
K 15 MICRO2 1M(01)
L 15 MICRO2 1M(01)
M 15 MICRO2 1M(01)
N 15 MICRO2 1M(01)
B 16 MICRO2 1M(01)
C 16 MICRO2 1M(01)
D 16 MICRO2 1M(01)
E 16 MICRO2 1M(01)
F 16 MICRO2 1M(01)
G 16 MICRO2 1M(01)
H 16 MICRO2 1M(01)
I 16 MICRO2 1M(01)
J 16 MICRO2 1M(01)
K 16 MICRO2 1M(01)
L 16 MICRO2 1M(01)

```
                                                                    B 1
                                        ;8207  .TOC "INTLOG.MIC"
                                        ;8208  .TOC "REVISION 26.0"
                                        ;8209  ;     Gerard Koeckhoven, C. E. MCDOWELL,   P. R. GUILBAULT
                                        ;8210
;8211    .NOBIN
;8212    .TOC     "      Revision History"
;8213
;8214    ; 26    Remove free location
;8215    ; 25    Documentation change
;8216    ;       Recover FREE.00C and FREE.00D, now used in IANDE.
;8217    ; 24    Reassign addresses in EDIV routine
;8218    ;       Restore micro-address in ADWC routine, which was previously swapped to fix EDIV problem.
;8219    ; 23    Assign permanent addresses to IL.EDIVC and IL.EDIVC1
;8220    ; 22    Swap a micro-address in ADWC routine to fix a bug in EDIV
;8221    ;       Fix EDIV problem such that largest negative number is acceptable.
;8222    ; 21    Fix two FPA interface bugs.
;8223    ;       Change FPA interface for integer multiply.
;8224    ; 20    Initial release.
                                        ;8225          .BIN
```

```
                                ;8226  .TOC    "      Integer, Logical, & Address     : BIT, CMP, TST"
                                ;8227
                                ;8228  ;*************************************************************************
                                ;8229  ;       BIT(B,W,L) mask.rx src.rx
                                ;8230  ;          Input (Dest is memory)       Q               Mask
                                ;8231  ;                                       MDR             Src
                                ;8232  ;          Input (Dest is register)     MDR             Mask
                                ;8233  ;                                       GPR(rnum)       Src
                                ;8234  ;*************************************************************************
                                ;8235
                                ;8236  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;8237  IL.BIT.B.W.L.MEM:                      ; 93 B3 D3
                                ;8238          ;-----------------------------;
                                ;8239          WB_M[MDR].AND.Q,CCOP2,       ;
U 000A, 0081,200A,0130,1047,003F,9  ;8240      SIZE[IDEP],IRD1              ;
                                ;8241
                                ;8242  IL.BIT.B.W.L.REG:                      ; 93 B3 D3
                                ;8243          ;-----------------------------;
                                ;8244          WB_M[MDR].AND.R[GPR.R],CCOP2, ;
U 004E, 0481,2002,013C,D047,003F,9  ;8245      SIZE[IDEP],IRD1              ;
                                ;8246
                                ;8247
                                ;8248
                                ;8249  ;*************************************************************************
                                ;8250  ;       CMP(B,W,L) src1.rx src2.rx
                                ;8251  ;          Input (Dest is memory)       Q               Source 1
                                ;8252  ;                                       MDR             Source 2
                                ;8253  ;          Input (Dest is register)     MDR             Source 1
                                ;8254  ;                                       GPR(rnum)       Source 2
                                ;8255  ;*************************************************************************
                                ;8256
                                ;8257  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;8258  IL.CMP.B.W.L.MEM:                      ; 91 B1 D1
                                ;8259          ;-----------------------------;
                                ;8260          WB_Q-M[MDR],CCOP1,SIZE[IDEP], ;
U 0052, 0481,200B,0130,0847,003F,9  ;8261      IRD1                         ;
                                ;8262
                                ;8263  IL.CMP.B.W.L.REG:                      ; 91 B1 D1
                                ;8264          ;-----------------------------;
                                ;8265          WB_M[MDR]-R[GPR.R],SIZE[IDEP], ;
U 006E, 0081,2000,013C,C847,003F,9  ;8266      CCOP1,IRD1                   ;
                                ;8267
                                ;8268
                                ;8269
                                ;8270  ;*************************************************************************
                                ;8271  ;       TST(B,W,L) src.rx
                                ;8272  ;          Input                        MDR     Source
                                ;8273  ;*************************************************************************
                                ;8274
                                ;8275  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;8276  IL.TST.B.W.L:                          ; 95 B5 D5
                                ;8277          ;-----------------------------;
U 008E, 0081,2592,4130,0847,003F,9  ;8278      WB_M[MDR],CCOP1,SIZE[IDEP],IRD1 ;
```

```
					.TOC	''	Integer, Logical, & Address	: CVT''

					;*********************************************************************
					;	CVT(LB,LW,WB) src.rx dst.wy
					;	   Input (Dest is memory)	Q	Source
					;					VA	Add of dest
					;	   Input (Dest is register)	MDR	Source
					;					RNUM	Register number of dest
					;*********************************************************************
					.REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
					IL.CVT.LB.LW.WB.MEM:			; F6 F7 33
						;-------------------------------;
						M[TEMPO]_Q,WRITE,CCOP1,		; WRT DEST & SAVE RESULT IN TEMPO
U 00AE, 0886,003A,48BD,8DD8,0049,C		SIZE[IDEP],IR<2>?		; GO CHECK FOR V

					.REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
					=0	;0------------------------------; CVTWB
						M[TEMPO]_SEXT(MB).XOR.Q,	; SIGN EXT RESULT & COMPARE WITH SOURCE
U 049C, 0886,081B,4030,0047,0040,F		SIZE[IDEP],NEXT/IL.CVTWB.STRIP	; (BITS<31:17> ARE GARBAGE)

						;1------------------------------; CVTLB CVTLW
						M[TEMPO]_SEXT(MB).XOR.Q,CCOP2,	; SIGN EXT RESULT & COMPARE WITH SOURCE
U 049D, 0[illegible],081B,4130,1047,003F,9		SIZE[IDEP],IRD1			; DO CCOP2 TO SET/CLR V

					.REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
					IL.CVT.LB.LW.WB.REG:			; F6 F7 33
						;-------------------------------;
						R[GPR.R].SIZ_Q_M[MDR],CCOP1,	; WRT DEST & SAVE RESULT IN Q
U 00CE, 0483,2592,58BC,C847,0049,E		SIZE[IDEP],IR<2>?		; GO CHECK FOR V

					.REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
					=0	;0------------------------------; CVTWB
						MTEMPO_SEXT(MDR).XOR.Q,		; SIGN EXT RESULT & COMPARE WITH SOURCE
U 049E, 0887,281B,4030,0047,0040,F		SIZE[IDEP],NEXT/IL.CVTWB.STRIP	; (BITS<31:17> ARE GARBAGE)

						;1------------------------------; CVTLB CVTLW
						MTEMPO_SEXT(MDR).XOR.Q,CCOP2,	; SIGN EXT RESULT & COMPARE WITH SOURCE
U 049F, 0087,281B,4130,1047,003F,9		SIZE[IDEP].IRD1			; DO CCOP2 TO SET/CLR V

					IL.CVTWB.STRIP:
						;-------------------------------;
						M[TEMPO]_ZEXT(MB),SIZE[WORD],	; ZEXT TO STRIP GARBAGE FROM <31:17>
U 040F, 0486,059E,4110,1047,003F,9		CCOP2,IRD1			; & DO CCOP2 TO SET/CLR V



					;*********************************************************************
					;	CVT(BW,BL,WL) src.rx dst.wy
					;	   Input			MDR	Source
					;*********************************************************************
					.REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
					IL.CVT.BW.BL.WL:			; 99 98 32
						;-------------------------------;
						Q_SEXT(M[MDR]),SIZE[IDEP],	; SAVE SEXT SRC IN Q. GO EVALUATE DEST
U 00E5, 0081,2816,51BD,8047,0014,0		LOD INC BRA?,NEXT/OS.WRT1	; RETURN THROUGH ROM TO DIVQUO+
```

```
                                        ;8334  .TOC     ""      Integer, Logical, & Address     : MOV(B,W,L), MOVA(B,W,L), MCOM, MNE
                                        ;8335
                                        ;8336  ;*****************************************************************************
                                        ;8337  ;       MOV (B,W,L)   src.rx dst.wy
                                        ;8338  ;       MOVA(B,W,L/F) src.ax dst.wl
                                        ;8339  ;       MCOM(B,W,L)   src.rx dst.wx
                                        ;8340  ;       MNEG(B,W,L)   src.rx dst.wx
                                        ;8341  ;          Input (Dest is memory)       Q       Source
                                        ;8342  ;                                       VA      Add of dest
                                        ;8343  ;          Input (Dest is register)     MDR     Source
                                        ;8344  ;                                       RNUM    Register number of dest
                                        ;8345  ;*****************************************************************************
                                        ;8346  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;8347  IL.MOV.B.W.L.MEM:                       ; 90 B0 D0
                                        ;8348  IL.MOVA.B.W.L.MEM:                      ; 9E 3E DE
                                        ;8349          ;-------------------------------;
                                        ;8350          R[DST.R],SIZ_Q_Q_D,WRITE NOTREG,; (ALSO USED BY MFPR)
U 00EE, 0882,002C,713C,55DA,003F,9      ;8351          SIZE[IDEP],CCOP2,IRD1           ;
                                        ;8352
                                        ;8353  IL.MOV.B.W.L.REG:                       ; 90 B0 D0
                                        ;8354  IL.MOVA.B.W.L.REG:                      ; 9E 3E DE
                                        ;8355          ;-------------------------------;
                                        ;8356          R[DST.R],SIZ_M[MDR],SIZE[IDEP], ;
U 00F1, 0883,2592,413C,55DA,003F,9      ;8357          WRITE NOTREG,CCOP2,IRD1         ;
                                        ;8358
                                        ;8359
                                        ;8360  IL.MCOM.B.W.L.MEM:                      ; 92 B2 D2
                                        ;8361          ;-------------------------------;
                                        ;8362          WRITE Q.NOT,CCOP2,              ;
U 01AF, 0080,00B8,013D,95D8,003F,9      ;8363          SIZE[IDEP],IRD1                 ;
                                        ;8364
                                        ;8365  IL.MCOM.B.W.L.REG:                      ; 92 B2 D2
                                        ;8366          ;-------------------------------;
                                        ;8367          R[GPR.R],SIZ_M[MDR].NOT,CCOP2,  ;
U 01BF, 0083,2E53,C13C,D047,003F,9      ;8368          SIZE[IDEP],IRD1                 ;
                                        ;8369
                                        ;8370
                                        ;8371  IL.MNEG.B.W.L.MEM:                      ; 8E AE CE
                                        ;8372          ;-------------------------------;
U 01CF, 0480,0038,013D,8DD8,003F,9      ;8373          WRITE -Q,CCOP1,SIZE[IDEP],IRD1  ;
                                        ;8374
                                        ;8375  IL.MNEG.B.W.L.REG:                      ; 8E AE CE
                                        ;8376          ;-------------------------------;
                                        ;8377          R[GPR.R],SIZ_-M[MDR],CCOP1,     ;
U 01D0, 0483,2593,013C,C847,003F,9      ;8378          SIZE[IDEP],IRD1                 ;
                                        ;8379
                                        ;8380
                                        ;8381  ;*****************************************************************************
                                        ;8382  ;       MOVZ(BW,BL,WL) src.rx dst.wy
                                        ;8383  ;          Input                        MDR     Source
                                        ;8384  ;*****************************************************************************
                                        ;8385  IL.MOVZ.BW.BL.WL:                       ; 9B 9A 3C
                                        ;8386          ;-------------------------------;
                                        ;8387          Q_ZEXT(M[MDR]),SIZE[IDEP],      ; STRIP GARB FRM UPPR BITS OF SRC, EVAL
U 01DE, 0481,2016,51BD,8047,0014,0      ;8388          LOD INC BRA?,NEXT/OS.WRT1       ; DEST, IRDX ROM RET TO MOV.B.W.L.MEM
```

```
                                    ;8389  .TOC     ""      Integer, Logical, & Address     : MOVQ, MOVAQ/D""
                                    ;8390
                                    ;8391  ;*************************************************************************
                                    ;8392  ;       MOVQ  src.rx dst.wy
                                    ;8393  ;          Input (Dest is memory)       TEMP1   LSB Source
                                    ;8394  ;                                       Q       MSB Source
                                    ;8395  ;                                       VA      Add of dest
                                    ;8396  ;          Input (Dest is register)     TEMP1   LSB Source
                                    ;8397  ;                                       MDR     MSB Source
                                    ;8398  ;                                       RNUM    Register number of dest
                                    ;8399  ;*************************************************************************
                                    ;8400  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                    ;8401  IL.MOVQ:                                ; 7D
                                    ;8402          ;-------------------------------;
                                    ;8403          R[DST.R]_M[TEMP1],WRITE NOTREG, ; WRITE LS WORD OF SOURCE
                                    ;8404          CCOP1,SIZE[IDEP],SET MM.NOINT,  ;
U 01E5, 0864,1592,493C,4DDA,004A,0  ;8405          REG MODE?                       ;
                                    ;8406
                                    ;8407  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                    ;8408  =0
                                    ;8409  IL.MOVQMEM:
                                    ;8410          ;0-------------------------------;
U 04A0, 0C80,0036,4030,0447,0041,F  ;8411          VA_VA+4,NEXT/IL.MOVQMEM.A       ;
                                    ;8412
                                    ;8413  IL.MOVQREG:
                                    ;8414          ;1-------------------------------;
                                    ;8415          R[DST.R+1]_M[MDR],WRITE NOTREG, ; WRITE MS WORD OF SOURCE
U 04A1, 0C85,2592,412F,55DA,003F,9  ;8416          CCOP2,SIZE[LONG],IRD1           ;
                                    ;8417
                                    ;8418  IL.MOVQMEM.A:
                                    ;8419          ;-------------------------------;
U 041F, 0080,003A,412D,95D8,003F,9  ;8420          WRITE Q,CCOP2,SIZE[LONG],IRD1   ; WRITE MS WORD OF SOURCE
                                    ;8421
                                    ;8422
                                    ;8423
                                    ;8424
                                    ;8425  ;*************************************************************************
                                    ;8426  ;       MOVAQ/D src.ax dst.wl
                                    ;8427  ;          Input (Dest is memory)       Q       Source
                                    ;8428  ;                                       VA      Add of dest
                                    ;8429  ;          Input (Dest is register)     MDR     Source
                                    ;8430  ;                                       RNUM    Register number of dest
                                    ;8431  ;*************************************************************************
                                    ;8432  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                    ;8433  IL.MOVA.Q.MEM:                          ; 7E
                                    ;8434          ;-------------------------------;
                                    ;8435          R[DST.R],SIZ_Q_Q_D,WRITE NOTREG,;
U 01EE, 0882,002C,713C,4DDA,003F,9  ;8436          CCOP1,SIZE[IDEP],IRD1           ;
                                    ;8437
                                    ;8438  IL.MOVA.Q.REG:                          ; 7E
                                    ;8439          ;-------------------------------;
                                    ;8440          R[DST.R],SIZ_M[MDR],SIZE[IDEP], ;
U 01F1, 0883,2592,413C,4DDA,003F,9  ;8441          WRITE NOTREG,CCOP1,IRD1         ;
```

```
                                        ;8442  .TOC    "       Integer, Logical, & Address     : PUSH, PUSHA"
                                        ;8443
                                        ;8444  ;*****************************************************************************
                                        ;8445  ;
                                        ;8446  ;       PUSHL src.rl
                                        ;8447  ;       PUSHA(B,W,L/F,Q/D) src.ax
                                        ;8448  ;          Input                              MDR     Source
                                        ;8449  ;
                                        ;8450  ;*****************************************************************************
                                        ;8451  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;8452  IL.PUSHL:                                    ; DD
                                        ;8453  IL.PUSHA.B.W.L:                              ; 9F 3F DF
                                        ;8454          ;-------------------------------;
U 0206, 0485,473C,0027,84A7,0042,F      ;8455          VA_R[SP]_RB-CONX(4),PUSH RBS-   ;
                                        ;8456
                                        ;8457  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        ;8458          ;-------------------------------;
                                        ;8459          WRITE M[MDR],SIZE[LONG],CCOP2,  ;
U 042F, 0481,2592,4120,15D8,003F,9      ;8460          IRD1                            ;
                                        ;8461
                                        ;8462
                                        ;8463
                                        ;8464  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;8465  IL.PUSHA.Q:                                 ; 7F
                                        ;8466          ;-------------------------------;
U 020D, 0C85,473C,0027,84A7,0044,1      ;8467          VA_R[SP]_RB-CONX(4),PUSH RBS-   ;
                                        ;8468
                                        ;8469  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        ;8470          ;-------------------------------;
                                        ;8471          WRITE M[MDR],SIZE[LONG],CCOP1,  ;
U 0441, 0481,2592,4120,0DD8,003F,9      ;8472          IRD1                            ;
```

```
                                        ;8473  .TOC    ""      Integer, Logical, & Address     : ROTL""
                                        ;8474
                                        ;8475  ;*****************************************************************************
                                        ;8476  ;
                                        ;8477  ;       ROTL cnt.rb src.rl dst.wl
                                        ;8478  ;          Input (Dest is memory)       Q       Count
                                        ;8479  ;                                       MDR     Source
                                        ;8480  ;                                       OSR     Dest
                                        ;8481  ;          Input (Dest is register)     Q       Count
                                        ;8482  ;                                       MDR     Source
                                        ;8483  ;                                       RNUM    Register number of dest
                                        ;8484  ;                                       OSR     Dest
                                        ;8485  ;
                                        ;8486  ;*****************************************************************************
                                        ;8487
                                        ;8488  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;8489  =1111
                                        ;8490  IL.ROTLMEM:                              ; 9C
                                        ;8491          ;1111-------------------------;
U 000F, 0080,0BFA,403D,8047,004A,2      ;8492          PL_Q                            ; LOAD PLATCH WITH COUNT
                                        ;8493
                                        ;8494  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        ;8495  =0      ;0-------------------------;
                                        ;8496          PUSH,R[TEMP1] M[MDR].RL.P,      ; ROL BY  CNT = ROR BY -CNT
                                        ;8497          CCOP2,SIZE[IDEP],               ;
U 04A2, 0885,2877,0630,5047,0414,0      ;8498          BRA ON ADD?,NEXT/OS.WRT1        ; GO EVALUATE DEST
                                        ;8499
                                        ;8500          ;1-------------------------;
U 04A3, 0080,1592,4130,05D8,003F,9      ;8501          WRITE M[TEMP1],SIZE[IDEP],IRD1  ; WRITE RESULT
                                        ;8502
                                        ;8503  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;8504  IL.ROTLREG:                              ; 9C
                                        ;8505          ;-------------------------;
U 0211, 0880,0BFA,403D,8047,0044,E      ;8506          PL_Q                            ; LOAD PLATCH WITH COUNT
                                        ;8507
                                        ;8508  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        ;8509          ;-------------------------;
                                        ;8510          R[GPR.R] M[MDR].RL.P,CCOP2,     ; ROL BY CNT = ROR BY -CNT
U 044E, 0085,2877,013C,D047,003F,9      ;8511          SIZE[IDEP],IRD1                 ;
```

```
                                        .TOC    ''      Integer, Logical, & Address     : CLR, ADWC''

                                        ;*****************************************************************************
                                        ;       CLR(B,W,L,Q) dst.wx
                                        ;          Input        VA/RNUM         Add/Reg of dest
                                        ;*****************************************************************************
                                        .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        IL.CLR.B.W.L:                                   ; 94 B4 D4
                                                ;---------------------------------------;
                                                R[DST.R].SIZ_0,WRITE NOTREG,            ;
U 0221, 0882,05B7,013C,55DA,003F,9              SIZE[IDEP],CCOP2,IRD1                   ;

                                        IL.CLRQ:                                        ; 7C
                                                ;---------------------------------------;
                                                R[DST.R]_0,WRITE NOTREG,                ; WRITE 0 TO FIRST WORD
                                                SIZE[IDEP],SET MM.NOINT,                ;
U 022A, 0064,05B7,093C,45DA,004A,4              REG MODE?                               ; & BUT ON REG MODE

                                        .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        =0      ;0--------------------------------------; CLRQMEM
U 04A4, 0480,0036,4030,0447,004A,5              VA_VA+4                                 ;

                                                ;1--------------------------------------; CRLQ(REG,MEM)
                                                R[DST.R+1]_0,WRITE NOTREG,CCOP1,        ; WRITE 0 TO SECOND WORD
U 04A5, 0C84,05B7,012F,4DDA,003F,9              SIZE[LONG],IRD1                         ;



                                        ;*****************************************************************************
                                        ;       ADWC add.rl sum.ml
                                        ;          Input (Dest is memory)       Q               Add
                                        ;                                       MDR             Sum
                                        ;                                       VA              Add of sum
                                        ;          Input (Dest is register)     MDR             Add
                                        ;                                       GPR(rnum)       Sum
                                        ;                                       RNUM            Register number of sum
                                        ;*****************************************************************************
                                        .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        IL.ADWCREG:                                     ; D8
                                                ;---------------------------------------;
                                                R[GPR.R]_M[MDR]+RB+PSLC,CCOP1,          ;
U 022E, 0085,20C1,013C,C847,003F,9              SIZE[IDEP],IRD1                         ;

                                        IL.ADWCMEM:                                     ; D8
                                                ;---------------------------------------;
                                                WRITE M[MDR]+Q+PSLC,                    ; DON'T SET CC IN CASE U TRAP
U 0231, 0061,20C9,0030,05D8,0045,F              SIZE[IDEP],SET MM.NOINT                 ;

                                        .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        045F:
                                                ;---------------------------------------;
                                                WB_M[MDR]+Q+PSLC,CCOP1,                 ; NO U TRAP SET CC
U 045F, 0081,20C9,0130,0847,003F,9              SIZE[IDEP],IRD1                         ;
```

```
                                        ;8565  .TOC    "       Integer, Logical, & Address     : ADD"
                                        ;8566
                                        ;8567  ;*************************************************************************
                                        ;8568  ;       ADD(B,W,L)2 add.rx sum mx
                                        ;8569  ;          Input (Dest is memory)       Q               Add
                                        ;8570  ;                                      MDR             Sum
                                        ;8571  ;                                      VA              Add of sum
                                        ;8572  ;          Input (Dest is register)     MDR             Add
                                        ;8573  ;                                      GPR(rnum)       Sum
                                        ;8574  ;                                      RNUM            Register number of sum
                                        ;8575  ;
                                        ;8576  ;       ADD(B,W,L)3 add1.rx add2.rx sum.wx
                                        ;8577  ;          Input (Dest is memory)       Q               Add1
                                        ;8578  ;                                      MDR             Add2
                                        ;8579  ;                                      OSR             Sum
                                        ;8580  ;          Input (Dest is register)     Q               Add1
                                        ;8581  ;                                      MDR             Add2
                                        ;8582  ;                                      RNUM            Register number of sum
                                        ;8583  ;                                      OSR             Sum
                                        ;8584  ;*************************************************************************
                                        ;8585
                                        ;8586  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;8587  IL.ADD2.B.W.L.REG:                    ; 80 A0 C0
                                        ;8588          ;-----------------------------;
                                        ;8589          R[GPR.R].SIZ_M[MDR]+RB,CCOP1, ;
U 0237, 0083,2001,013C,C847,003I,9      ;8590          SIZE[IDEP],IRD1               ;
                                        ;8591
                                        ;8592  IL.ADD2.B.W.L.MEM:                    ; 80 A0 C0
                                        ;8593  IL.ADD3.B.W.L.REG:                    ; 81 A1 C1
                                        ;8594          ;-----------------------------;
                                        ;8595          R[DST.R].SIZ_M[MDR]+Q,CCOP1,  ;
U 023D, 0883,2009,013C,4DDA,003F,9      ;8596          WRITE NOTREG,SIZE[IDEP],IRD1  ;
                                        ;8597  =00000
                                        ;8598  =01111
                                        ;8599  IL.ADD3.B.W.L.MEM:                    ; 81 A1 C1
                                        ;8600          ;01111--------------------------;
                                        ;8601          PUSH,R[TEMP1].SIZ_M[MDR]+Q,   ;
                                        ;8602          CCOP1,SIZE[IDEP],             ;
U 004F, 0C83,2009,0630,4847,0414,0      ;8603          BRA ON ADD?,NEXT/OS.WRT1      ;
                                        ;8604
                                        ;8605          ;10000--------------------------;
U 0050, 0080,1592,4130,05D8,003F,9      ;8606          WRITE M[TEMP1],SIZE[IDEP],IRD1 ;
                                        ;8607  =
```

```
                                ;8608  .TOC   "      Integer, Logical, & Address     : INC, ADAWI"
                                ;8609
                                ;8610  ;*****************************************************************************
                                ;8611  ;       INC(LB,LW,WB) sum.mx
                                ;8612  ;          Input                                MDR             Sum
                                ;8613  ;                                               VA/RNUM         Add/Reg of sum
                                ;8614  ;*****************************************************************************
                                ;8615
                                ;8616  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;8617  IL.INC.B.W.L:                                  ; 96 B6 D6
                                ;8618          ;-------------------------------;
                                ;8619          R[DST.R].SIZ_M[MDR]+1,CCOP1,    ;
U 0243, 0C83,2E50,013C,4DDA,003F,9  ;8620      WRITE NOTREG,SIZE[IDEP],IRD1    ;
                                ;8621
                                ;8622
                                ;8623  ;*****************************************************************************
                                ;8624  ;       ADAWI add.rx sum.mx
                                ;8625  ;          Input (Dest is memory)       Q               Add
                                ;8626  ;                                       MDR             Sum
                                ;8627  ;          Input (Dest is register)     MDR             Add
                                ;8628  ;                                       GPR(rnum)       Sum
                                ;8629  ;                                       RNUM            Register number of sum
                                ;8630  ;*****************************************************************************
                                ;8631
                                ;8632  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;8633  IL.ADAWIREG:                                 ; 58
                                ;8634          ;-------------------------------;
                                ;8635          R[GPR.RJ.SIZ_M[MDR]+RB,         ;
U 0247, 0083,2001,013C,D047,003F,9  ;8636      SIZE[IDEP],CCOP2,IRD1           ;
                                ;8637
                                ;8638  IL.ADAWIMEM:                               ; 58
                                ;8639          ;-------------------------------;
U 024E, 0881,B002,42BD,8047,004A,6  ;8640      VA<0>?                          ; ALLIGNED ?
                                ;8641
                                ;8642  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                ;8643  =0      ;0-------------------------------; ALLIGNED
                                ;8644          READ.MOD.LOCK,SIZE[IDEP],       ; READ SUM & SET LOCK
U 04A6, 0480,0036,4030,0053,0046,2  ;8645      NEXT/IL.ADAWIMEM2               ;
                                ;8646
                                ;8647          ;1-------------------------------; UN-ALLIGNED
U 04A7, 0C80,0036,4030,0047,00FF,8  ;8648      NEXT/IE.OPER.FAULT              ; RESERVED OPERAND
                                ;8649
                                ;8650  IL.ADAWIMEM2:
                                ;8651          ;-------------------------------;
                                ;8652          WRITE.UL M[MDR]+Q,CCOP2,        ;
U 0462, 0081,2009,0130,15D8,003F,9  ;8653      SIZE[IDEP],IRD1                 ;
```

```
                                       .TOC    "      Integer, Logical, & Address     : SUB"

                                       ;*****************************************************************************
                                       ;        SUB(B,W,L)2 sub.rx dif.mx
                                       ;           Input  (Dest is memory)        Q              Sub
                                       ;                                          MDR            Dif
                                       ;                                          VA             Add of dif
                                       ;           Input  (Dest is register)      MDR            Sub
                                       ;                                          GPR(rnum)      Dif
                                       ;                                          RNUM           Register number of dif
                                       ;
                                       ;        SUB(B,W,L)3 sub.rx min.rx dif.wx
                                       ;           Input  (Dest is memory)        Q              Sub
                                       ;                                          MDR            Min
                                       ;                                          OSR            Dif
                                       ;           Input  (Dest is register)      Q              Sub
                                       ;                                          MDR            Min
                                       ;                                          RNUM           Register number of dif
                                       ;                                          OSR            Dif
                                       ;*****************************************************************************

                                       .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                       IL.SUB2.B.W.L.REG:                       ; 82 A2 C2
                                               ;--------------------------------;
                                               R[GPR.R].SIZ_RB-M[MDR],CCOP2,    ;
U 0251, 0483,2003,013C,D047,003F,9             SIZE[IDEP],IRD1                  ;

                                       IL.SUB2.B.W.L.MEM:                       ; 82 A2 C2
                                       IL.SUB3.B.W.L.REG:                       ; 83 A3 C3
                                               ;--------------------------------;
                                               R[DST.R].SIZ_M[MDR]-Q,CCOP2,     ;
U 025B, 0C83,2008,013C,55DA,003F,9             WRITE NOTREG,SIZE[IDEP],IRD1     ;
                                       =00000
                                       =01111
                                       IL.SUB3.B.W.L.MEM:                       ; 83 A3 C3
                                               ;01111---------------------------;
                                               PUSH,R[TEMP1].SIZ_M[MDR]-Q,      ;
                                               CCOP2,SIZE[IDEP],                ;
U 006F, 0883,2008,C630,5047,0414,0             BRA ON ADD?.NEXT/OS.WRT1         ;

                                               ;10000---------------------------;
U 0070, 0080,1592,4130,05D8,003F,9             WRITE M[TEMP1],SIZE[IDEP],IRD1   ;
                                       =
```

```
                                ;8697  .TOC    "       Integer, Logical, & Address     : DEC, SBWC"
                                ;8698
                                ;8699  ;*************************************************************************
                                ;8700  ;       DEC(LB,LW,WB) dif.mx
                                ;8701  ;          Input                                MDR             Dif
                                ;8702  ;                                               VA/RNUM         Add/Reg of sum
                                ;8703  ;*************************************************************************
                                ;8704
                                ;8705  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;8706  IL.DEC.B.W.L:                                  ; 97 B7 D7
                                ;8707          ;-------------------------------;
                                ;8708          R[DST.R].SIZ_M[MDR]-1,CCOP1,    ;
U 0265, 0883,2E51,013C,4DDA,003F,9 ;8709       WRITE NOTREG,SIZE[IDEP],IRD1    ;
                                ;8710
                                ;8711
                                ;8712
                                ;8713  ;*************************************************************************
                                ;8714  ;       SBWC sub.rl dif.ml
                                ;8715  ;          Input (Dest is memory)       Q               Sub
                                ;8716  ;                                       MDR             Dif
                                ;8717  ;                                       VA              Add of dif
                                ;8718  ;          Input (Dest is register)     MDR             Sub
                                ;8719  ;                                       GPR(rnum)       Dif
                                ;8720  ;                                       RNUM            Register no. of dif
                                ;8721  ;*************************************************************************
                                ;8722
                                ;8723  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;8724  IL.SBWCREG:                                  ; D9
                                ;8725          ;-------------------------------;
                                ;8726          R[GPR.R] R8-M[MDR]-PSLC,CCOP1,  ;
J 026E, 0485,20C3,013C,C847,003F,9 ;8727       SIZE[IDEP],IRD1                 ;
                                ;8728
                                ;8729  IL.SBWCMEM:                                  ; D9
                                ;8730          ;-------------------------------;
                                ;8731          WRITE M[MDR]-Q-PSLC,SIZE[IDEP], ; DON'T SET CC IN CASE U TRAP
J 0271, 0061,30C8,0030,05D8,0046,4 ;8732       SET MM.NOINT                    ;
                                ;8733
                                ;8734  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                ;8735          ;-------------------------------;
                                ;8736          WB M[MDR]-Q-PSLC,CCOP1,         ; DON'T SET CC IN CASE U TRAP
J 0464, 0481,20C8,0130,0847,003F,9 ;8737       SIZE[IDEP],IRD1                 ;
```

```
                                      .TOC    "       Integer, Logical, & Address     : XOR"

                                      ;******************************************************************************
                                      ;       XOR(B,W,L)2 mask.rx dst.mx
                                      ;          Input (Dest is memory)        Q              Mask
                                      ;                                        MDR            Dst
                                      ;                                        VA             Add of Dst
                                      ;          Input (Dest is register)      MDR            Mask
                                      ;                                        GPR(rnum)      Dst
                                      ;                                        RNUM           Register number of Dst
                                      ;
                                      ;       XOR(B,W,L)3 mask.rx src.rx dst.mx
                                      ;          Input (Dest is memory)        Q              Mask
                                      ;                                        MDR            Src
                                      ;                                        OSR            Dst
                                      ;          Input (Dest is register)      Q              Mask
                                      ;                                        MDR            Src
                                      ;                                        RNUM           Register number of dst
                                      ;                                        OSR            Dst
                                      ;******************************************************************************

                                      .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                      IL.XOR2.B.W.L.REG:                            ; 8C AC CC
                                              ;-------------------------------------;
                                              R[GPR.R].SIZ_M[MDR].XOR.RB,           ;
U 0277, 0083,2003,413C,D047,003F,9            SIZE[IDEP],CCOP2,IRD1                 ;

                                      IL.XOR2.B.W.L.MEM:                            ; 8C AC CC
                                      IL.XOR3.B.W.L.REG:                            ; 8D AD CD
                                              ;-------------------------------------;
                                              R[DST.R].SIZ_M[MDR].XOR.Q,            ;
                                              WRITE NOTREG,CCOP2,SIZE[IDEP],        ;
U 0279, 0883,2008,413C,55DA,003F,9            IRD1                                  ;
                                      =00000
                                      =01111
                                      IL.XOR3.B.W.L.MEM:                            ; 8D AD CD
                                              ;01111--------------------------------;
                                              PUSH,R[TEMP1].SIZ_M[MDR].XOR.Q,       ;
                                              CCOP2,SIZE[IDEP],                     ;
U 008F, 0083,2008,4630,5047,0414,0            BRA ON ADD?,NEXT/OS.WRT1              ;

                                              ;10000--------------------------------;
U 0090, 0080,1592,4130,05D8,003F,9            WRITE M[TEMP1],SIZE[IDEP],IRD1        ;
                                      =
```

```
                                        .TOC    "       Integer, Logical, & Address     : BIS"

                                        ;***************************************************************************
                                        ;       BIS(B,W,L)2 mask.rx dst.mx
                                        ;          Input (Dest is memory)        Q               Mask
                                        ;                                        MDR             Dst
                                        ;                                        VA              Add of Dst
                                        ;          Input (Dest is register)      MDR             Mask
                                        ;                                        GPR(rnum)       Dst
                                        ;                                        RNUM            Register number of Dst
                                        ;
                                        ;       BIS(B,W,L)3 mask.rx src.rx dst.mx
                                        ;          Input (Dest is memory)        Q               Mask
                                        ;                                        MDR             Src
                                        ;                                        OSR             Dst
                                        ;          Input (Dest is register)      Q               Mask
                                        ;                                        MDR             Src
                                        ;                                        RNUM            Register number of dst
                                        ;                                        OSR             Dst
                                        ;***************************************************************************

                                        .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        IL.BIS2.B.W.L.REG:                      ; 88 A8 C8
                                                ;-------------------------------;
                                                R[GPR.R].SIZ_M[MDR].OR.RB,      ;
U 027F, 0483,2002,413C,D047,003F,9              SIZE[IDEP],CCOP2,IRD1           ;

                                        IL.BIS2.B.W.L.MEM:                      ; 88 A8 C8
                                        IL.BIS3.B.W.L.REG:                      ; 89 A9 C9
                                                ;-------------------------------;
                                                R[DST.R].SIZ_M[MDR].OR.Q,       ;
                                                WRITE NOTREG,CCOP2,SIZE[IDEP],  ;
U 028B, 0C83,200A,413C,55DA,003F,9              IRD1                            ;
                                        =00000
                                        =01111
                                        IL.BIS3.B.W.L.MEM:                      ; 89 A9 C9
                                                ;01111--------------------------;
                                                PUSH,R[TEMP1].SIZ_M[MDR].OR.Q,  ;
                                                CCOP2,SIZE[IDEP],               ;
U 00AF, 0883,200A,4630,5047,0414,0              BRA ON ADD?,NEXT/OS.WRT1        ;

                                                ;10000--------------------------;
U 00B0, 0080,1592,4130,05D8,003F,9              WRITE M[TEMP1],SIZE[IDEP],IRD1  ;
                                        =
```

```
                                ;8826  .TOC    "       Integer, Logical, & Address     : BIC"
                                ;8827
                                ;8828  ;*************************************************************************
                                ;8829  ;       BIC(B,W,L)2 mask.rx dst.mx
                                ;8830  ;          Input (Dest is memory)       Q               Mask
                                ;8831  ;                                       MDR             Dst
                                ;8832  ;                                       VA              Add of Dst
                                ;8833  ;          Input (Dest is register)     MDR             Mask
                                ;8834  ;                                       GPR(rnum)       Dst
                                ;8835  ;                                       RNUM            Register number of Dst
                                ;8836  ;
                                ;8837  ;       BIC(B,W,L)3 mask.rx src.rx dst.mx
                                ;8838  ;          Input (Dest is memory)       Q               Mask
                                ;8839  ;                                       MDR             Src
                                ;8840  ;                                       OSR             Dst
                                ;8841  ;          Input (Dest is register)     Q               Mask
                                ;8842  ;                                       MDR             Src
                                ;8843  ;                                       RNUM            Register number of dst
                                ;8844  ;                                       OSR             Dst
                                ;8845  ;*************************************** ***********************************
                                ;8846
                                ;8847  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;8848  IL.BIC2.B.W.L.REG:                      ; 8A AA CA
                                ;8849          ;-------------------------------;
                                ;8850          R[GPR.R].SIZ_M[MDR].NOTAND.RB,  ;
U 028E, 0483,2003,C13C,D047,003F,9  ;8851      SIZE[IDEP],CCOP2,IRD1           ;
                                ;8852
                                ;8853  IL.BIC2.B.W.L.MEM:                      ; 8A AA CA
                                ;8854  IL.BIC3.B.W.L.REG:                      ; 8B AB CB
                                ;8855          ;-------------------------------;
                                ;8856          R[DST.R].SIZ_M[MDR].ANDNOT.Q,   ;
                                ;8857          WRITE NOTREG,CCOP2,SIZE[IDEP],  ;
U 0291, 0883,200B,813C,55DA,003F,9  ;8858      IRD1                            ;
                                ;8859  =00000
                                ;8860  =01111
                                ;8861  IL.BIC3.B.W.L.MEM:                      ; 8B AB CB
                                ;8862          ;01111--------------------------;
                                ;8863          PUSH,                           ;
                                ;8864          R[TEMP1].SIZ_M[MDR].ANDNOT.Q,   ;
                                ;8865          CCOP2,SIZE[IDEP],               ;
U 00CF, 0C83,200B,8630,5047,0414,0  ;8866      BRA ON ADD?,NEXT/OS.WRT1        ;
                                ;8867
                                ;8868          ;10000--------------------------;
U 00D0, 0080,1592,4130,05D8,003F,9  ;8869      WRITE M[TEMP1],SIZE[IDEP],IRD1  ;
                                ;8870  =
```

```
                                    ;8871  .TOC    "       Integer, Logical, & Address     : MUL without FPA"
                                    ;8872
                                    ;8873  ;*************************************************************************
                                    ;8874  ;       MUL(B,W,L)2 mulr.rx prod.mx
                                    ;8875  ;          Input        Q                Mulr
                                    ;8876  ;                       MDR              Prod
                                    ;8877  ;                       VA               Add of prod
                                    ;8878  ;
                                    ;8879  ;       MUL(B,W,L)3 mulr.rx muld.rx prod.wx
                                    ;8880  ;          Input        Q                Mulr
                                    ;8881  ;                       MDR              Muld
                                    ;8882  ;                       OSR              Muld
                                    ;8883  ;
                                    ;8884  ;          Resources    TEMP4   Multiplier
                                    ;8885  ;                       TEMP6   Sext Multiplicand
                                    ;8886  ;                       FLAG0   Product will be neg (MULSUB when +- or -+)
                                    ;8887  ;                       D       MSB of double length prod(right justified)
                                    ;8888  ;                       Q       LSB of double length prod(right justified)
                                    ;8889  ;                       MDR     0
                                    ;8890  ;
                                    ;8891  ;          Subroutines  MULSUB.MDR_0
                                    ;8892  ;*************************************************************************
                                    ;8893
                                    ;8894  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                    ;8895  IL.MUL2.B.W.L:                          ; 84 A4 C4
                                    ;8896          ;-------------------------------;
                                    ;8897          R[TEMP6]_SEXT(M[MDR]),          ; TEMP6 <- CAND. ALUS<0> <- SIGN
                                    ;8898          ALUS_SIGND,SIZE[IDEP],          ;
U 0297, 0C85,2E5E,0031,98C7,0004,1  ;8899          NEXT/IL.MULA                    ;
                                    ;8900  =000
                                    ;8901  IL.MUL3.B.W.L:                          ; 84 A4 C4
                                    ;8902          ;000-----------------------------;
                                    ;8903          PUSH,R[TEMP6]_SEXT(M[MDR]),     ; TEMP6 <- CAND
                                    ;8904          ALUS_SIGND,SIZE[IDEP],          ; ALUS<0> <- SIGN
U 0040, 0485,2E5E,01B1,98C7,0414,0  ;8905          LOD INC BRA?,NEXT/OS.WRT1       ; GO EVALUATE PROD
                                    ;8906
                                    ;8907  IL.MULA:
                                    ;8908          ;001-----------------------------;
                                    ;8909          PUSH,M[TEMP4]_Q,SIZE[IDEP],     ; TEMP4 <- LIER.  BRANCH ON SIGN
U 0041, 0486,403A,4B7D,88C7,0403,0  ;8910          CMP SIGNS?,NEXT/IL.MULSUB.MDR_0 ; COMPARE TO MULTIPLY SUB
                                    ;8911  =100
                                    ;8912          ;100-----------------------------; RETURN+3 PROD WILL BE +
                                    ;8913          R[DST.R].SIZ_M[MDR].OR.Q,       ; WRITE PROD & SET CC(MDR=0)
                                    ;8914          WRITE NOTREG,SIZE[IDEP],        ;
                                    ;8915          SET MM.NOINT,CCOP1 SIGND?,      ; BRANCH ON SIGN TO SET/CLR V
U 0044, 0063,200A,4B7C,4DDA,0044,5  ;8916          NEXT/IL.MULD                    ;
                                    ;8917
                                    ;8918          ;101-----------------------------; RETURN+4 PROD WILL BE -
                                    ;8919          WB_D+ALKC,                      ; D+ALKC = 0 THEN NO V
U 0045, 0C80,0061,0A3D,8047,0845,2 351* ;8920      WX.EQ.0?,NEXT/IL.MULC           ; ELSE V OR MOST NEG NO.
                                    ;8921  =
```

```
                                        ;8922  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        ;8923  =00
                                        ;8924  IL.MULB:
                                        ;8925          ;00-----------------------------; D.EQ.0 : V OR MOST NEG
                                        ;8926          WB_(R[ZERO]+Q).SL.1,SIZE[IDEP], ; Q = MOST NEG?
U 0450, 0880,0039,CDBD,8047,0845,1 391* ;8927          WX(SIZ).EQ.0?                   ;
                                        ;8928
                                        ;8929          ;01-----------------------------; (D.NE.0) OR (Q.NE.MOST NEG) : V
                                        ;8930          R[DST.R].SIZ_M[MDR]-Q,CCOP1,    ; MDR IS 0
                                        ;8931          WRITE NOTREG,SIZE[IDEP],        ;
U 0451, 0863,2008,003C,4DDA,0044,7      ;8932          SET MM.NOINT,NEXT/IL.MULSETV    ;
                                        ;8933
                                        ;8934  IL.MULC:
                                        ;8935          ;10-----------------------------; (D+ALKC).NE.0 : V OR MOST NEG NO
                                        ;8936          WB_D,                           ; D.NE.0 IS V, D.EQ.0 THEN CHK Q FOR
U 0452, 0880,0022,4A7D,8047,0045,0      ;8937          WX.NE.0?,NEXT/IL.MULB           ; MOST NEG NO.
                                        ;8938
                                        ;8939  IL.MULWRITE-Q:
                                        ;8940          ;11-----------------------------; ((D+ALKC).EQ.0) OR (Q.EQ.MOST NEG)
                                        ;8941          R[DST.R].SIZ_M[MDR]-Q,CCOP1,    ; MDR IS 0
U 0453, 0C83,2008,013C,4DDA,003F,9      ;8942          WRITE NOTREG,SIZE[IDEP],IRD1    ;
                                        ;8943  =01
                                        ;8944  IL.MULD:
                                        ;8945          ;01-----------------------------; Q IS POS LOOK AT D FOR V
U 0445, 0480,0022,413D,9047,003F,9      ;8946          WB_D,SIZE[IDEP],CCOP2,IRD1      ;
                                        ;8947
                                        ;8948  IL.MULSETV:
                                        ;8949          ;11-----------------------------;
U 0447, 0480,0036,4130,08E7,003F,9      ;8950          SET V,IRD1                      ;
```

```
                                        ;8951  .TOC    ""      Integer, Logical, & Address     : MUL with FPA""
                                        ;8952
                                        ;8953  ;*****************************************************************************
                                        ;8954  ;       MUL(B,W,L)2 mulr.rx prod.mx
                                        ;8955  ;          Input (Dest is memory)       Q               Mulr
                                        ;8956  ;                                       MDR             Prod
                                        ;8957  ;                                       VA              Add of prod
                                        ;8958  ;          Input (Dest is register)     MDR             Mulr
                                        ;8959  ;                                       GPR(rnum)       Prod
                                        ;8960  ;                                       RNUM            Register number of Prod
                                        ;8961  ;
                                        ;8962  ;       MUL(B,W,L)3 mulr.rx muld.rx prod.wx
                                        ;8963  ;          Input (Dest is memory)       Q               Mulr
                                        ;8964  ;                                       MDR             Muld
                                        ;8965  ;          Input (Dest is register)     Q               Mulr
                                        ;8966  ;                                       MDR             Muld
                                        ;8967  ;*****************************************************************************
                                        ;8968  .REGION/IRD1.R1L,IRD1.R1H
                                        ;8969
                                        ;8970  FI.MUL2.B.W.L.MEM:
                                        ;8971          ;-------------------------------;
U 02A7, 0C81,2036,4030,0047,1001,D      ;8972          FPA_M[MDR],NEXT/FI.MULSTALL.PROD; 2ND OP ON MB, STALL FOR FPA & FINISH
                                        ;8973  =0
                                        ;8974  FI.MUL3.B.W.L.MEM:
                                        ;8975          ;0------------------------------;
                                        ;8976          PUSH,FPA_M[MDR],                ; WRITE 2ND OP
U 001C, 0481,2036,41B0,0047,1414,0      ;8977          LOD INC BRA?,NEXT/OS.WRT1       ; GET OP3
                                        ;8978
                                        ;8979  FI.MULSTALL.PROD:
                                        ;8980          ;1------------------------------;
                                        ;8981          FPAWAIT,R[TEMP1]_FPA,ALUS_SIGND,; WAIT FOR RESULT, SAVE SIGN
U 001D, 0E64,0036,4030,58C7,407F,0      ;8982          SIZE[IDEP],SET MM.NOINT
                                        ;8983
                                        ;8984  07F0:   ;*********FORCE ADDRESS*********;
                                        ;8985          R[DST.R].SIZ_M[TEMP1],          ; WRITE RESULT AND SET CCODES
U 07F0, 0882,1592,403C,4DDA,007F,1      ;8986          WRITE NOTREG,SIZE[IDEP],CCOP1
                                        ;8987
                                        ;8988  07F1:   ;*********FORCE ADDRESS*********; SPLIT ON SIGN OF RESULT
U 07F1, 0684,0036,4B70,5847,40FE,D      ;8989          FPAWAIT,R[TEMP1]_FPA,ALUS?      ; GET UPPER HALF OF RESULT
                                        ;8990
                                        ;8991  0FED:   ;01******FORCE ADDRESS*********; RESULT IS POSITIVE, NO OVERFLOW IF
U 0FED, 0080,05BE,4130,5047,003F,9      ;8992          WB_R[TEMP1],CCOP2,SIZE[IDEP],IRD1 ;UPPER HALF IS 0.
                                        ;8993
                                        ;8994  0FEF:   ;11******FORCE ADDRESS*********; RESULT IS NEGATIVE, NO OVERFLOW IF
                                        ;8995          WB_R[TEMP1]+1,CCOP2,SIZE[IDEP], ; UPPER HALF IS -1.
U 0FEF, 0480,0E7C,0130,5047,003F,9      ;8996          IRD1                            ;
```

```
                                ;8997  .TOC    ""       Integer, Logical, & Address     : EMUL without FPA""
                                ;8998
                                ;8999  ;*************************************************************************
                                ;9000  ;       EMUL mulr.rl muld.rl add.rl prod.wq
                                ;9001  ;          Input        Q                Mulr
                                ;9002  ;                       MDR              Muld
                                ;9003  ;                       OSR              Muld
                                ;9004  ;
                                ;9005  ;          Resources    TEMP6   Sext Multiplicand
                                ;9006  ;                       TEMP4   Multiplier
                                ;9007  ;                       TEMP2   MSB Addend/Work
                                ;9008  ;                       TEMP1   LSB Addend
                                ;9009  ;                       FLAG0   Product will be neg (MULSUB when +- or -+)
                                ;9010  ;                       D       MSB of double length prod(right justified)
                                ;9011  ;                       Q       LSB of double length prod(right justified)
                                ;9012  ;       Subroutines     MULSUB.MDR_0
                                ;9013  ;*************************************************************************
                                ;9014  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;9015  =00
                                ;9016  Il.EMUL:                                  ; 7A
                                ;9017          ;00-------------------------------;
                                ;9018          PUSH,M[TEMP4]_Q,                 ; TEMP4 <- LIER
U 0004, 0086,403A,41BD,8047,0410,0 ;9019       LOD INC BRA?,NEXT/OS.RED         ; EVALUATE ADDEND
                                ;9020
                                ;9021          ;01-------------------------------;
                                ;9022          PUSH,M[TEMP6]_Q,                 ; TEMP6 <- CAND. ALUS<0> <- SIGN
                                ;9023          ALUS_SIGND,SIZE[IDEP],           ;
U 0005, 0C86,603A,41BD,98C7,0415,0 ;9024       LOD INC BRA?,NEXT/OS.WRT2        ; EVALUATE PROD
                                ;9025
                                ;9026          ;10-------------------------------;
U 0006, 0C86,103A,4B6D,8047,0044,D ;9027       M[TEMP1]_Q,SIZE[LONG],SIGND CMP?; TEMP1 <- LSB OF ADD. ADD POS OR NEG?
                                ;9028  =
                                ;9029  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                ;9030  =01     ;01-------------------------------;
U 044D, 0084,05B7,0030,8047,0044,0 ;9031       R[TEMP2]_0,NEXT/IL.EMUL1         ; ADDEND IS POS. SET MSB TO 0
                                ;9032
                                ;9033          ;11-------------------------------;
U 044F, 0484,0E77,0030,8047,0044,0 ;9034       R[TEMP2]_-1,NEXT/IL.EMUL1        ; ADDEND IS NEG. SET MSB TO -1
                                ;9035  =000
                                ;9036  IL.EMUL1:
                                ;9037          ;000------------------------------;
                                ;9038          PUSH,WB_Q M[TEMP4],SIZE[LONG],   ; TEMP4 <- LIER. BRANCH ON SIGN
U 0440, 0080,4002,5B6D,88C7,0403,0 ;9039       CMP SIGNS?,NEXT/IL.MULSUB.MDR_0  ; COMPARE TO MULTIPLY SUB
                                ;9040  =011
                                ;9041  FI.EMUL.ADD:
                                ;9042          ;011------------------------------; RETURN+3 PROD IS POS OR EMUL WITH FPA
                                ;9043          R[DST.R]_M[TEMP1]+Q,SIZE[IDEP],  ; WRITE LSB OF PRODUCT
                                ;9044          WRITE NOTREG,CCOP1,SET_MM.NOINT,;
U 0443, 0064,1009,093C,4DDA,004A,8 ;9045       REG MODE?,NEXT/IL.EMUL2          ;
                                ;9046
                                ;9047  =100    ;100------------------------------; RETURN+4 PROD IS NEG
                                ;9048          R[DST.R]_M[TEMP1]-Q,SIZE[IDEP],  ; WRITE LSB OF PRODUCT
                                ;9049          WRITE NOTREG,CCOP1,SET_MM.NOINT,;
U 0444, 0C64,1008,093C,4DDA,004A,A ;9050       REG MODE?,NEXT/IL.EMUL3          ;
                                ;9051  =
```

```
                                        ;9052  =0
                                        ;9053  IL.EMUL2:
                                        ;9054          ;0--------------------------------; DEST IS MEM
U 04A8, 0480,1009,0030,0447,004A,9      ;9055          WB_M[TEMP1]+Q,VA_VA+4            ; DO ADD TO RESTORE ALKC & BUMP VA
                                        ;9056
                                        ;9057          ;1--------------------------------; DEST IS REG (OR MEM FROM ABOVE)
                                        ;9058          R[TEMP2] D+RB+ALKC,CCOP2,        ; TEMP2 <- MSB OF PROD, SET CC
U 04A9, 0884,0061,0030,9047,0046,6      ;9059          SIZE[IDEP],NEXT/IL.EMUL4         ; GO WRITE MSB PROD
                                        ;9060  =0
                                        ;9061  IL.EMUL3:
                                        ;9062          ;0--------------------------------; DEST IS MEM
U 04AA, 0880,1008,0030,0447,004A,B      ;9063          WB_M[TEMP1]-Q,VA_VA+4            ; DO SUB TO RESTORE ALKC & BUMP VA
                                        ;9064
                                        ;9065          ;1--------------------------------; DEST IS REG (OR MEM FROM ABOVE)
                                        ;9066          R[TEMP2] RB-D-ALKC,CCOP2,        ; TEMP2 <- MSB OF PROD, SET CC
U 04AB, 0C84,0063,0030,9047,0046,6      ;9067          SIZE[IDEP]                       ;
                                        ;9068
                                        ;9069  IL.EMUL4:
                                        ;9070          ;--------------------------------;
                                        ;9071          R[DST.R+1] M[TEMP2],             ; WRITE MSB OF PROD
U 0466, 0C84,2592,412F,45DA,003F,9      ;9072          WRITE NOTREG,SIZE[LONG],IRD1     ;
```

```
                                ;9073  .TOC    ""      Integer, Logical, & Address     : EMUL with FPA""
                                ;9074
                                ;9075  ;*************************************************************************************
                                ;9076  ;       EMUL mulr.rl muld.rl add.rl prod.wq
                                ;9077  ;          Input        Q                Mulr
                                ;9078  ;                       MDR              Muld
                                ;9079  ;                       OSR              Muld
                                ;9080  ;
                                ;9081  ;          Resources    TEMP3   Work
                                ;9082  ;                       TEMP2   MSB Addend/Work
                                ;9083  ;                       TEMP1   LSB Addend
                                ;9084  ;                       TEMP0   Work
                                ;9085  ;                       D       MSB of double length prod
                                ;9086  ;                       Q       LSB of double length prod
                                ;9087  ;
                                ;9088  ;*************************************************************************************
                                ;9089
                                ;9090  .REGION/IRD1.R1L,IRD1.R1H
                                ;9091  =00
                                ;9092  FI.EMUL:
                                ;9093          ;00-------------------------------;
                                ;9094          LOD INC BRA?,                     ; GET THE ADD.RL
U 0018, 0480,0036,4180,0047,0410,0 ;9095       PUSH,NEXT/OS.RED                  ; AND WRITE MULD.RL TO FPA
                                ;9096
                                ;9097          ;01-------------------------------;
                                ;9098          PUSH,R[TEMP1]_M[MDR],             ; ADD.RL TO WORKING TEMP
                                ;9099          ALUS_SIGND,SIZE[IDEP],            ;
U 0019, 0085,2592,4180,5807,0414,0 ;9100       LOD INC BRA?,NEXT/OS.WRT1         ; SAVE SIGN AND GET PROD.WQ
                                ;9101
                                ;9102          ;10-------------------------------;
                                ;9103          R[TEMP2]_0,                       ; GUESS SECOND HALF = 0
U 001A, 0084,05B7,0B70,9847,0000,9 ;9104       ALUS?                             ; CORRECT?
                                ;9105
                                ;9106  =
                                ;9107  =01
                                ;9108  FI.EMUL.10:
                                ;9109          ;01-------------------------------; POSITIVE OR ZERO
                                ;9110          FPAWAIT,                          ; STALL FOR FPA
                                ;9111          R[TEMP0]_FPA,                     ; SAVE LSB OF PRODUCT
U 0009, 0284,0036,4030,0047,402A,E ;9112       NEXT/FI.EMUL.20                   ; CONTINUE ON
                                ;9113
                                ;9114          ;11-------------------------------; NEGATIVE
                                ;9115          R[TEMP2]_-1,                      ; RESET OTHER HALF
J 000B, 0484,0E77,0030,8047,0000,9 ;9116       NEXT/FI.EMUL.10                   ; CONTINUE ON
                                ;9117
                                ;9118  FI.EMUL.20:
                                ;9119          ;-------------------------------;
                                ;9120          FPAWAIT,                          ; WAIT FOR FPA
U 02AE, 0A84,0036,4030,C047,402B,1 ;9121       R[TEMP3]_FPA                      ; GET MSB OF PRODUCT
                                ;9122
                                ;9123          ;-------------------------------;
                                ;9124          Q_M[TEMP0] D_R[TEMP3],            ; GET TO CORRECT D AND Q
U 02B1, 0480,0005,7030,C047,0044,3 ;9125       NEXT/FI.EMUL.ADD                  ; JOIN MAIN FLOWS OF EMUL
```

```
                                        ;9126  .TOC    ""      Integer, Logical, & Address     : MULSUB.MDR_0""
                                        ;9127
                                        ;9128  ;*************************************************************************
                                        ;9129  ;       Inputs          TEMP4   Multiplier
                                        ;9130  ;                       TEMP6   Sext Multiplicand
                                        ;9131  ;                       DSIZE   No. of bits to mul(0=8, 1=16, 2=32, 3=32)
                                        ;9132  ;                       LATCH
                                        ;9133  ;
                                        ;9134  ;       Outputs         D       MSB of double length prod (right justified)
                                        ;9135  ;                       Q       LSB of double length prod (right justified)
                                        ;9136  ;                       FLAG0   Product will be neg (when +- or -+)
                                        ;9137  ;                       MDR     0
                                        ;9138  ;
                                        ;9139  ;       Returns         +3      Product will be pos
                                        ;9140  ;                       +4      Product will be neg
                                        ;9141  ;
                                        ;9142  ;       Resources       STEPC
                                        ;9143  ;*************************************************************************
                                        ;9144  =00
                                        ;9145  IL.MULSUB.MDR_0:
                                        ;9146          ;00-----------------------------; MULSUB CAND+ LIER+
                                        ;9147          Q_SEXT(M[TEMP4]),               ; Q <- SIGN EXTENDED LIER, MDR <- 0
                                        ;9148          SIZE[IDEP],MDR_0,               ;
U 0030, 0880,4816,5C7D,84E7,0004,8      ;9149          DSIZE?,NEXT/IL.MULSUB.CAND+     ;
                                        ;9150
                                        ;9151          ;01-----------------------------; MULSUB CAND+ LIER-
                                        ;9152          Q_-SEXT(M[TEMP4]),              ; Q <- SIGN EXTENDED LIER, MDR <- 0
                                        ;9153          SIZE[IDEP],MDR_0,SET FLAG0,     ; SET PROD WILL BE NEG FLAG
U 0031, 0840,4817,1C7D,84E7,0004,8      ;9154          DSIZE?,NEXT/IL.MULSUB.CAND+     ;
                                        ;9155
                                        ;9156          ;10-----------------------------; MULSUB CAND- LIER+
                                        ;9157          Q_SEXT(M[TEMP4]),               ; Q <- SIGN EXTENDED LIER, MDR <- 0
                                        ;9158          SIZE[IDEP],MDR_0,SET FLAG0,     ; SET PROD WILL BE NEG FLAG
U 0032, 0040,4816,5C7D,84E7,0005,4      ;9159          DSIZE?,NEXT/IL.MULSUB.CAND-     ;
                                        ;9160
                                        ;9161          ;11-----------------------------; MULSUB CAND- LIER-
                                        ;9162          Q_-SEXT(M[TEMP4]),              ; Q <- SIGN EXTENDED LIER, MDR <- 0
                                        ;9163          SIZE[IDEP],MDR_0,               ;
U 0033, 0080,4817,1C7D,84E7,0005,4      ;9164          DSIZE?,NEXT/IL.MULSUB.CAND-     ;
```

```
                                    ;9165  =00
                                    ;9166  IL.MULSUB.CAND+:
                                    ;9167          ;00-------------------------------; MULSUB CAND+ BYTE
                                    ;9168          MULFAST+ CAND IN R[TEMP6],       ;
                                    ;9169          SIZE[IDEP],STEPC 2,              ;
U 0048, 00A0,0027,9031,8047,0004,6  ;9170          NEXT/IL.MULSUB.+LOOP             ;
                                    ;9171
                                    ;9172          ;01-------------------------------; MULSUB CAND+ WORD
                                    ;9173          MULFAST+ CAND IN R[TEMP6],       ;
                                    ;9174          SIZE[IDEP],STEPC 6,              ;
U 0049, 08A8,0027,9031,8047,0004,6  ;9175          NEXT/IL.MULSUB.+LOOP             ;
                                    ;9176
                                    ;9177          ;10-------------------------------; MULSUB CAND+ LONG
                                    ;9178          MULFAST+ CAND IN R[TEMP6],       ;
                                    ;9179          SIZE[IDEP],STEPC 14.,            ;
U 004A, 08B0,0027,9031,8047,0004,6  ;9180          NEXT/IL.MULSUB.+LOOP             ;
                                    ;9181
                                    ;9182          ;11-------------------------------; MULSUB CAND+ QUAD(EMUL)
                                    ;9183          MULFAST+ CAND IN R[TEMP6],       ;
                                    ;9184          SIZE[IDEP],STEPC 14.,            ;
U 004B, 08B0,0027,9031,8047,0004,6  ;9185          NEXT/IL.MULSUB.+LOOP             ;
                                    ;9186  =0
                                    ;9187  IL.MULSUB.+LOOP:
                                    ;9188          ;0--------------------------------;
                                    ;9189          MULFAST+ CAND IN R[TEMP6],       ;
                                    ;9190          SIZE[IDEP],                      ;
U 0046, 0880,0027,9331,8047,0004,6  ;9191          DBZ STEPC?,NEXT/IL.MULSUB.+LOOP  ; STAY IN LOOP TILL STEP CNT = 0
                                    ;9192
                                    ;9193          ;1--------------------------------;
                                    ;9194          MULFAST+ CAND IN R[TEMP6],       ;
U 0047, 0480,0027,9431,A047,0004,C  ;9195          ISIZE[IDEP],FLAG0?               ; SC = 0. WILL PROD BE POS OR NEG?
                                    ;9196  =0
                                    ;9197          ;0--------------------------------;
                                    ;9198          MULFAST+ CAND IN R[TEMP6],       ;
U 004C, 0480,0027,90B1,8047,0000,3  ;9199          SIZE[IDEP],RETURN [+3]           ; PROD WILL BE POS
                                    ;9200
                                    ;9201          ;1--------------------------------;
                                    ;9202          MULFAST+ CAND IN R[TEMP6],       ;
U 004D, 0C80,0027,90B1,8047,0000,4  ;9203          SIZE[IDEP],RETURN [+4]           ; PROD WILL BE NEG
```

```
                                        ;9204   =00
                                        ;9205   IL.MULSUB.CAND-:
                                        ;9206           ;00-------------------------------; MULSUB CAND- BYTE
                                        ;9207           MULFAST- CAND IN R[TEMP6],      ;
                                        ;9208           SIZE[IDEP],STEPC_2,             ;
U 0054, 0CA0,0026,9031,8047,0006,C      ;9209           NEXT/IL.MULSUB.-LOOP            ;
                                        ;9210
                                        ;9211           ;01-------------------------------; MULSUB CAND- WORD
                                        ;9212           MULFAST- CAND IN R[TEMP6],      ;
                                        ;9213           SIZE[IDEP],STEPC_6,             ;
U 0055, 04A8,0026,9031,8047,0006,C      ;9214           NEXT/IL.MULSUB.-LOOP            ;
                                        ;9215
                                        ;9216           ;10-------------------------------; MULSUB CAND- LONG
                                        ;9217           MULFAST- CAND IN R[TEMP6],      ;
                                        ;9218           SIZE[IDEP],STEPC_14.,           ;
U 0056, 04B0,0026,9031,8047,0006,C      ;9219           NEXT/IL.MULSUB.-LOOP            ;
                                        ;9220
                                        ;9221           ;10-------------------------------; MULSUB CAND- QUAD(EMUL)
                                        ;9222           MULFAST- CAND IN R[TEMP6],      ;
                                        ;9223           SIZE[IDEP],STEPC_14.,           ;
U 0057, 04B0,0026,9031,8047,0006,C      ;9224           NEXT/IL.MULSUB.-LOOP            ;
                                        ;9225   =0
                                        ;9226   IL.MULSUB.-LOOP:
                                        ;9227           ;0--------------------------------;
                                        ;9228           MULFAST- CAND IN R[TEMP6],      ;
                                        ;9229           SIZE[IDEP],                     ;
U 006C, 0480,0026,9331,8047,0006,C      ;9230           DBZ STEPC?,NEXT/IL.MULSUB.-LOOP ; STAY IN LOOP TILL STEP CNT = 0
                                        ;9231
                                        ;9232           ;1--------------------------------;
                                        ;9233           MULFAST- CAND IN R[TEMP6],      ;
U 006D, 0080,0026,9431,8047,0007,2      ;9234           SIZE[IDEP],FLAG0?               ; SC = 0. WILL PROD BE POS OR NEG?
                                        ;9235   =0
                                        ;9236           ;0--------------------------------;
                                        ;9237           MULFAST- CAND IN R[TEMP6],      ;
U 0072, 0080,0026,90B1,8047,0000,3      ;9238           SIZE[IDEP],RETURN [+3]          ; PROD WILL BE POS
                                        ;9239
                                        ;9240           ;1--------------------------------;
                                        ;9241           MULFAST- CAND IN R[TEMP6],      ;
U 0073, 0880,0026,90B1,8047,0000,4      ;9242           SIZE[IDEP],RETURN [+4]          ; PROD WILL BE NEG
```

```
                                ;9243  .TOC    ""       Integer, Logical, & Address     : DIV""
                                ;9244
                                ;9245  ;*************************************************************************************
                                ;9246  ;       DIV(B,W,L)2 divr.rx quo.mx
                                ;9247  ;          Input        Q                 Divr
                                ;9248  ;                       MDR               Quo
                                ;9249  ;                       VA                Add of quo
                                ;9250  ;
                                ;9251  ;       DIV(B,W,L)3 divr.rx divd.rx quo.wx
                                ;9252  ;          Input        Q                 Divr
                                ;9253  ;                       MDR               Divd
                                ;9254  ;                       OSR               Divd
                                ;9255  ;
                                ;9256  ;          Resources    TEMP3             Sext Divisor
                                ;9257  ;                       TEMP1             LSB Dividend
                                ;9258  ;                       TEMP0             0
                                ;9259  ;                       FLAG3             Divisor Pos (Set by DIVSUB)
                                ;9260  ;                       FLAG2             Ediv (MBZ)
                                ;9261  ;                       FLAG0             Quotient will be Neg (WHEN +- OR -+)
                                ;9262  ;                       D                 Uncorrected Remainder
                                ;9263  ;                       Q                 Quotient
                                ;9264  ;          Subroutines  DIVSUB
                                ;9265  ;*************************************************************************************
                                ;9266  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;9267  IL.DIVB2:
                                ;9268          ;-------------------------------; 86
                                ;9269          R[TEMP3] SEXT(M[MDR]),STEPC_2,  ; TEMP3 <- SIGN EXTENDED SOR. SET UP
                                ;9270          ALUS_SIGND,SIZE[IDEP],          ; CNT FOR BYTE. GO EVALUATE END(QUO).
U 02B9, 04A5,2E5E,01B0,D8C7,0011,0 ;9271          LOD INC BRA?,NEXT/OS.MOD        ; (IRD ROM WILL GO TO IL.DIV2.B.W.L)
                                ;9272  IL.DIVW2:
                                ;9273          ;-------------------------------; A6
                                ;9274          R[TEMP3] SEXT(M[MDR]),STEPC_6,  ; TEMP3 <- SIGN EXTENDED SOR. SET UP
                                ;9275          ALUS_SIGND,SIZE[IDEP],          ; CNT FOR BYTE. GO EVALUATE END(QUO).
U 02BF, 0CAD,2E5E,01B0,D8C7,0011,0 ;9276          LOD INC BRA?,NEXT/OS.MOD        ; (IRD ROM WILL GO TO IL.DIV2.B.W.L)
                                ;9277  IL.DIVL2:
                                ;9278          ;-------------------------------; C6
                                ;9279          R[TEMP3] SEXT(M[MDR]),STEPC_14.,; TEMP3 <- SIGN EXTENDED SOR. SET UP
                                ;9280          ALUS_SIGND,SIZE[IDEP],          ; CNT FOR BYTE. GO EVALUATE END(QUO).
U 02C9, 0CB5,2E5E,01B0,D8C7,0011,0 ;9281          LOD INC BRA?,NEXT/OS.MOD        ; (IRD ROM WILL GO TO IL.DIV2.B.W.L)
                                ;9282
                                ;9283  IL.DIVB3:
                                ;9284          ;-------------------------------; 87
                                ;9285          R[TEMP3] SEXT(M[MDR]),STEPC_2,  ; TEMP3 <- SIGN EXTENDED SOR. SET UP
                                ;9286          ALUS_SIGND,SIZE[IDEP],          ; CNT FOR BYTE. GO EVALUATE END(QUO).
U 02CD, 0CA5,2E5E,01B0,D8C7,0010,0 ;9287          LOD INC BRA?,NEXT/OS.RED        ; (IRD ROM WILL GO TO IL.DIV3.B.W.L)
                                ;9288  IL.DIVW3:
                                ;9289          ;-------------------------------; A7
                                ;9290          R[TEMP3] SEXT(M[MDR]),STEPC_6,  ; TEMP3 <- SIGN EXTENDED SOR. SET UP
                                ;9291          ALUS_SIGND,SIZE[IDEP],          ; CNT FOR BYTE. GO EVALUATE END(QUO).
U 0301, 04AD,2E5E,01B0,D8C7,0010,0 ;9292          LOD INC BRA?,NEXT/OS.RED        ; (IRD ROM WILL GO TO IL.DIV3.B.W.L)
                                ;9293  IL.DIVL3:
                                ;9294          ;-------------------------------; C7
                                ;9295          R[TEMP3] SEXT(M[MDR]),STEPC_14.,; TEMP3 <- SIGN EXTENDED SOR. SET UP
                                ;9296          ALUS_SIGND,SIZE[IDEP],          ; CNT FOR BYTE. GO EVALUATE END(QUO).
U 0305, 04B5,2E5E,01B0,D8C7,0010,0 ;9297          LOD INC BRA?,NEXT/OS.RED        ; (IRD ROM WILL GO TO IL.DIV3.B.W.L)
```

```
                                        ;9298  IL.DIV2.B.W.L:
                                        ;9299          ;-------------------------------;
                                        ;9300          MTEMP0_D_0_Q_SEXT(MDR),         ; Q< SIGN EXT END(LSB). D<- 0(MSB END)
U 0300, 0C87,2815,7B7D,9847,0045,6      ;9301          SIZE[IDEP],ALUS?,NEXT/IL.DIVFORK; CHK FOR DIV BY 0
                                        ;9302  =0
                                        ;9303  IL.DIV3.B.W.L:
                                        ;9304          ;0------------------------------;
                                        ;9305          PUSH,Q_SEXT(M[MDR]),SIZE[IDEP], ; Q <- SIGN EXT END(LSB)
U 008C, 0881,2816,51BD,8047,0414,0      ;9306          LOD INC BRA?,NEXT/OS.WRT1       ; GO EVALUATE QUO
                                        ;9307
                                        ;9308          ;1------------------------------;
                                        ;9309          M[TEMP0]_D_R[ZERO],             ; D <- 0(MSB END)
U 008D, 0886,05BE,6B7D,9847,0045,6      ;9310          ALUS?,NEXT/IL.DIVFORK           ; CHECK FOR DIV BY 0
                                        ;9311
                                        ;9312  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        ;9313  =00     ;00-----------------------------; RETURE-2 QUO WILL BE +
                                        ;9314          Q IS NEG?,SIZE[IDEP],           ; CHK QUO FOR V
U 0454, 0C80,003A,4B7D,8047,0005,1      ;9315          NEXT/IL.DIVQUO+                 ;
                                        ;9316
                                        ;9317          ;01-----------------------------; RETURN-1 QUO WILL BE -
                                        ;9318          R[DST.R].SIZ_M[TEMP0]-Q,CCOP1,  ; WRITE -Q (TEMP0=0)
U 0455, 0C82,0008,013C,4DDA,003F,9      ;9319          WRITE NOTREG,SIZE[IDEP],IRD1    ;
                                        ;9320
                                        ;9321  IL.DIVFORK:
                                        ;9322          ;10-----------------------------; SOR.NE.0
                                        ;9323          PUSH,M[TEMP1]_Q,SIZE[IDEP],     ; TEMP1 <- LSB END. BRANCH ON SIGN
U 0456, 0C86,103A,4B7D,88C7,0444,8      ;9324          CMP SIGNS?,NEXT/IL.DIVSUB       ; COMPARE TO DIVIDE SUB
                                        ;9325
                                        ;9326          ;11-----------------------------; SOR.EQ.0
                                        ;9327          R[DST.R].SIZ_M[TEMP0].OR.Q,     ; WRITE LSB END(TEMP0=0)
                                        ;9328          CCOP1,WRITE NOTREG,SIZE[IDEP],  ; AND GO TO CHARLIE
U 0457, 0062,000A,403C,4DDA,00FB,8      ;9329          SET MM.NOINT,NEXT/IF INT.DBZ    ;
                                        ;9330
                                        ;9331  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;9332  =01
                                        ;9333  IL.DIVQUO+:
                                        ;9334          ;01-----------------------------; QUO IS POS
                                        ;9335          R[DST.R].SIZ_Q_Q_D,SIZE[IDEP],  ; WRITE DEST(USED BY CVTBW.BL WL)
U 0051, 0882,002C,713C,4DDA,003F,9      ;9336          WRITE NOTREG,CCOP1,IRD1         ;
                                        ;9337
                                        ;9338          ;11-----------------------------; QUO IS NEG : OVERFLOW
                                        ;9339          R[DST.R].SIZ_M[TEMP1],CCOP1,    ; WRITE END
                                        ;9340          WRITE NOTREG,SIZE[IDEP],        ;
U 0053, 0862,1592,403C,4DDA,0044,7      ;9341          SET MM.NOINT,NEXT/IL.MULSETV    ; & GO SET V
```

```
                                ;9342  .TOC    "       Integer, Logical, & Address     : EDIV"
                                ;9343
                                ;9344  ;*************************************************************************
                                ;9345  ;       EDIV divr.rl divd.rq quo.wl rem.wl
                                ;9346  ;          Input        Q                 Divr
                                ;9347  ;                       MDR               Divd
                                ;9348  ;
                                ;9349  ;          Resources    TEMP7             RNUM/VA for Quotient OS
                                ;9350  ;                       TEMP5             Uncorrected Remainder
                                ;9351  ;                       TEMP4             RNUM/VA for Remainder OS
                                ;9352  ;                       TEMP3             Sext Divisor
                                ;9353  ;                       TEMP2             MSB Dividend
                                ;9354  ;                       TEMP1             LSB Dividend
                                ;9355  ;                       TEMP0             0
                                ;9356  ;                       FLAG3             Divisor Pos (Set by DIVSUB)
                                ;9357  ;                       FLAG2             Ediv (set to 1 by Ediv)
                                ;9358  ;                       FLAG1             OS for Quotient is reg mode
                                ;9359  ;                       FLAG0             Quotient will be Neg (WHEN +- OR -+)
                                ;9360  ;                       D                 Uncorrected Remainder
                                ;9361  ;                       Q                 Quotient
                                ;9362  ;                       PLATCH            Sign of Dividend, Divide By 0, Overflow
                                ;9363  ;                                         Dividend +      PL<4:0>=31      PL<5>=1
                                ;9364  ;                                         Dividend -      PL<4:0>=0       PL<5>=1
                                ;9365  ;                                         Divide BY 0     PL<4:0>=0       PL<5>=0
                                ;9366  ;                                         Overflow        PL<4:0>=1       PL<5>=0
                                ;9367  ;          Subroutines  DIVSUB
                                ;9368  ;                       MM.PRB.WRITE.SIZ
                                ;9369  ;*************************************************************************
                                ;9370
                                ;9371  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;9372
                                ;9373  =00
                                ;9374  IL.EDIV:
                                ;9375          ;00-------------------------------; 7B
                                ;9376          PUSH,R[TEMP2]_D_M[MDR],           ; TEMP2 & D <- MSB END
                                ;9377          STEPC 14.,                        ;
U 0064, 0CB5,2593,61B0,8047,0414,0  ;9378          LOD INC BRA?,NEXT/OS.WRT1         ; GO EVALUATE QUO
                                ;9379
                                ;9380          ;01-------------------------------;
                                ;9381          PUSH,M[TEMP3]_Q,ALUS_SIGND,       ; TEMP3 <- SOR. SET EDIV FLAG
                                ;9382          SIZE[IDEP],SET FLAG2,             ;
U 0065, 0456,303A,493D,98C7,044A,C  ;9383          REG MODE?,NEXT/IL.EDIVA           ; REGISTER MODE ?
                                ;9384
                                ;9385
                                ;9386          ;10-------------------------------; RETURN FROM OS.WRT1 FOR REM
                                ;9387          Q_R[TEMP1] M[TEMP0]_0,            ; Q <- LSB END. TEMP0_0 FOR DIVSUB
U 0066, 0486,05BC,7B70,5847,003F,6  ;9388          ALUS?,NEXT/IL.EDIVB               ; MDR <- 0 FOR WRT Q, BRA ON SOR = 0
                                ;9389  =
```

```
                                        ;9390  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        ;9391
                                        ;9392  =0
                                        ;9393  IL.EDIVA:
                                        ;9394          ;0------------------------------; NOT REG MODE
                                        ;9395          RTEMP7_M[VA] PL<4-0>_31 PL<5>_1,; SAVE VA, SET UP PL FOR DIVEND+,
U 04AC, 0185,BED2,01BF,F847,0014,0      ;9396          LOD INC BRA?,NEXT/OS.WRT1       ; AND GO EVALUATE REM
                                        ;9397
                                        ;9398          ;1------------------------------; REG MODE
                                        ;9399          RTEMP7_RNUM PL<4-0>_31 PL<5>_1, ; SAVE RNUM, SET UP PL FOR DIVIDEND+,
                                        ;9400          SET FLAG1,                      ; SET REG MODE FLAG,
U 04AD, 094D,6EF6,41BF,F847,0014,0      ;9401          LOD INC BRA?,NEXT/OS.WRT1       ; GO EVALUATE REM
                                        ;9402
                                        ;9403  03F4:
                                        ;9404          ;00****FORCE ADDRESS**********; RETURN-2 QUO WILL BE +
U 03F4, 0080,00B8,06FD,8047,083F,1 400* ;9405          Q.NOT IS POS?,NEXT/IL.EDIVC     ; CHECK FOR V
                                        ;9406
                                        ;9407  03F5:
                                        ;9408          ;01****FORCE ADDRESS**********; RETURN-1 QUO WILL BE
                                        ;9409          Q_R[ZERO]-Q,SIZE[LONG],         ; NEGATE QUOTIENT & SEE IF
U 03F5, 0480,0038,16ED,8047,083F,1 400* ;9410          WB<31-30>?,NEXT/IL.EDIVC        ; V OCCURED
                                        ;9411
                                        ;9412  03F6:
                                        ;9413  IL.EDIVB:
                                        ;9414          ;10****FORCE ADDRESS**********; SOR.NE.0
                                        ;9415          PUSH,WB_M[TEMP2],SIZE[LONG],    ; WBUS <- MSB DIVIDEND
U 03F6, 0480,2592,4B60,08C7,0C44,8 340* ;9416          CMP SIGNS?,NEXT/IL.DIVSUB       ; BRANCH ON SIGN COMPORE TO DIVSUB
                                        ;9417
                                        ;9418  03F7:
                                        ;9419          ;11****FORCE ADDRESS**********; SOR.EQ.0(DIVIDE BY 0)
U 03F7, 0180,0EF6,4030,0047,0046,C      ;9420          PL_[0],NEXT/IL.EDIVD            ; PL<4:0>=0, PL<5>=0 : DIVIDE BY 0
                                        ;9421  045A:
                                        ;9422  FREE.045A:
                                        ;9423          ;****LOCATION NOT USED*******; SOR.NE.0
                                        ;9424          PUSH,WB_M[TEMP2],SIZE[LONG],    ; WBUS <- MSB DIVIDEND
U 045A, 0480,2592,4B60,08C7,0C44,8 340* ;9425          CMP SIGNS?,NEXT/IL.DIVSUB       ; BRANCH ON SIGN COMPORE TO DIVSUB
                                        ;9426
                                        ;9427  045B:
                                        ;9428  FREE.045B:
                                        ;9429          ;****LOCATION NOT USED*******; SOR.EQ.0(DIVIDE BY 0)
U 045B, 018C,0EF6,4030,0047,0046,C      ;9430          PL_[0],NEXT/IL.EDIVD            ; PL<4:0>=0, PL<5>=0 : DIVIDE BY 0
                                        ;9431
                                        ;9432  0458:
                                        ;9433  FREE.0458:
                                        ;9434          ;00****LOCATION NOT USED******; RETURN-2 QUO WILL BE +
U 0458, 0480,00B8,86FD,8047,083F,1 355* ;9435          NOT.Q<31>?,NEXT/IL.EDIVC        ; CHECK FOR V
                                        ;9436
                                        ;9437  0459:
                                        ;9438  FREE.0459:
                                        ;9439          ;01****LOCATION NOT USED******; RETURN-1 QUO WILL BE
                                        ;9440          Q_R[ZERO]-Q,SIZE[LONG],         ; NEGATE QUOTIENT & SEE IF
U 0459, 0480,0038,1B6D,8047,083F,1 436* ;9441          SIGND CMP?,NEXT/IL.EDIVC        ; V OCCURED
```

```
                                        ;9442  03F1:
                                        ;9443  IL.EDIVC:
                                        ;9444          ;01-------------------------------; V    : (POS:Q<31> = 1)
U 03F1, 0880,003A,5A3D,8047,003F,2      ;9445          Q IS ZERO?,NEXT/IL.EDIVC1        ; V?   : (NEG: -Q = 0 OR POS )
                                        ;9446  03F2:
                                        ;9447  IL.EDIVC1:
                                        ;9448          ;10-------------------------------; OVERFLOW, Q NOT EQUAL TO ZERO
U 03F2, 0580,0EF6,4030,0847,0046,C      ;9449          PL_[1],NEXT/IL.EDIVD             ; PL<4:0>=1, PL<5>=0 :
                                        ;9450
                                        ;9451  03F3:
                                        ;9452          ;11-------------------------------; NO V : (NEG:-Q < 0), (POS:Q IS POS)
                                        ;9453          M[TEMP5]_D_REM,                  ; UN-SHIFT REM & SAVE IN TEMP6 & D
U 03F3, 0486,5026,AB7D,9847,0046,1      ;9454          ALUS?,NEXT/IL.EDIVF              ; TEST ALUS<1> FOR REM CORRECTION
                                        ;9455
                                        ;9456  045C:
                                        ;9457  FREE.045C:
                                        ;9458          ;00****LOCATION NOT USED********; V    : (POS:Q<31> = 1)
U 045C, 0580,0EF6,4030,0847,0046,C      ;9459          PL_[1],NEXT/IL.EDIVD             ; PL<4:0>=1, PL<5>=0 :
                                        ;9460
                                        ;9461  045D:
                                        ;9462  FREE.045D:
                                        ;9463          ;01****LOCATION NOT USED********; FREE LOCATION
                                        ;9464          M[TEMP5]_D_REM,                  ; UN-SHIFT REM & SAVE IN TEMP6 & D
U 045D, 0486,5026,AB7D,9847,0046,1      ;9465          ALUS?,NEXT/IL.EDIVF              ; TEST ALUS<1> FOR REM CORRECTION
                                        ;9466
                                        ;9467  045E:
                                        ;9468  FREE.045E:
                                        ;9469          ;11****LOCATION NOT USED********; NO V : (NEG:-Q < 0), (POS:Q IS POS)
                                        ;9470          M[TEMP5]_D_REM,                  ; UN-SHIFT REM & SAVE IN TEMP6 & D
U 045E, 0486,5026,AB7D,9847,0046,1      ;9471          ALUS?,NEXT/IL.EDIVF              ; TEST ALUS<1> FOR REM CORRECTION
                                        ;9472
                                        ;9473
                                        ;9474  IL.EDIVD:
                                        ;9475          ;-------------------------------; V OR DIVIDE BY 0
                                        ;9476          Q_M[TEMP1] D_R[ZERO],            ; Q <- LSB END. D <- 0
U 046C, 0080,1005,703D,8047,0046,1      ;9477          NEXT/IL.EDIVF                    ;
                                        ;9478  =00
                                        ;9479  IL.EDIVE:
                                        ;9480          ;00-------------------------------; SOR IS NEG RE-DO CORRECTION
U 0460, 0480,5000,2030,C047,0046,1      ;9481          D_M[TEMP5]-R[TEMP3]              ; D <- UNCORRECTED REM - SOR
                                        ;9482
                                        ;9483  IL.EDIVF:
                                        ;9484          ;01-------------------------------; ALL BS WITH D&Q IS DONE LETS MOVE ON
                                        ;9485          R[TEMP4]_RNUM,                   ; SAVE REG NO &
U 0461, 0085,6036,4931,0047,004A,E      ;9486          REG MODE?,NEXT/IL.EDIVG          ; SEE IF REG MODE
                                        ;9487  =11
                                        ;9488          ;11-------------------------------; CORRECT REMAINDER (ASSUME SOR POS)
                                        ;9489          D_D+R[TEMP3],FLAG3?,             ; D<-(UNCORRECTED REM-SOR) IS SOR POS
U 0463, 0880,0021,24F0,C047,0046,0      ;9490          NEXT/IL.EDIVE                    ;
```

```
                                        ;9491  =0
                                        ;9492  IL.EDIVG:
                                        ;9493          ;0------------------------------; NOT REG MODE
                                        ;9494          PUSH,R[TEMP4] M[VA],            ; SAVE VA & PROBE WRITE ACCESS
U 04AE, 0885,B592,4031,0047,0579,6      ;9495          NEXT/MM.PRB.WRITE.SIZ.00        ;
                                        ;9496
                                        ;9497          ;1------------------------------; REG MODE OR REG FROM GUO PROBE
U 04AF, 0881,D5BE,45F1,C4A7,0046,5      ;9498          VA_RNUM_R[TEMP7],FLAG1?         ; GET VA OR RNUM OF QUO, CHK REG MODE
                                        ;9499
                                        ;9500  =01     ;01------------------------------; QUO OP IS NOT REG MODE
                                        ;9501          WRITE Q,CCOP1,SET MM.NOINT,     ; WRITE QUOTIENT
U 0465, 086U,003A,403D,8DD8,0046,E      ;9502          SIZE[IDEP],NEXT/IL.EDIVH        ;
                                        ;9503
                                        ;9504          ;11------------------------------; QUO OP IS REG MODE
                                        ;9505          R[GPR.R].SIZ_M[TEMP0].OR.Q,     ; WRITE QUOTIENT (TEMP0=0)
U 0467, 0C62,000A,403C,C847,0046,E      ;9506          SIZE[IDEP],CCOP1,SET MM.NOINT   ;
                                        ;9507
                                        ;9508  IL.EDIVH:
                                        ;9509          ;------------------------------; RESTORE RNUN/VA OF REM OP & CHECK
                                        ;9510          VA_RNUM_R[TEMP4] CLOBBER Q,     ; RESTORE VA/RNUM(WHITCHEVER)
U 046E, 0081,DA04,7DF1,04A7,0046,8      ;9511          PL<4-0>.EQ.0? PL<5>?            ; OVERFLOW? END+? DIVBY0? END-?
                                        ;9512
                                        ;9513  =00     ;00------------------------------; V
                                        ;9514          R[DST.R] D,WRITE NOTREG,         ; WRITE REMAINDER(REM IS 0)
U 0468, 0484,05B2,403C,45DA,0044,7      ;9515          SIZE[IDEP],NEXT/IL.MULSETV      ; GO SET V
                                        ;9516
                                        ;9517          ;01------------------------------; DIVEDEND POSITIVE
                                        ;9518          R[DST.R] D,WRITE NOTREG,         ; WRITE REMAINDER
U 0469, 0884,05B2,413C,45DA,003F,9      ;9519          SIZE[IDEP],IRD1                 ;
                                        ;9520
                                        ;9521          ;10------------------------------; DIVIDE BY 0
                                        ;9522          R[DST.R] D,WRITE NOTREG,         ; WRITE REMAINDER(REM IS 0)
U 046A, 0C84,05B2,403C,45DA,00FB,8      ;9523          SIZE[IDEP],NEXT/IE.INT.DBZ      ; GO TO DIVIDE BY 0 ROUTINE
                                        ;9524
                                        ;9525          ;11------------------------------; DIVIDEND NEGATINE
                                        ;9526          R[DST.R] -D,WRITE NOTREG,        ; WRITE -REMAINDER
U 046B, 0884,05B3,013C,45DA,003F,9      ;9527          SIZE[IDEP],IRD1                 ;
```

```
                                        ;9528  .TOC    "       Integer, Logical, & Address     : DIVSUB"
                                        ;9529
                                        ;9530  ;*************************************************************************
                                        ;9531  ;       Inputs          D       MSB Dividend(MBZ when not EDIV)
                                        ;9532  ;                       Q       LSB Dividend
                                        ;9533  ;                       TEMP3   Sext Divisor
                                        ;9534  ;                       TEMP0   0
                                        ;9535  ;                       FLAG2   Ediv(0=Divide is not ediv, 1=divide is ediv)
                                        ;9536  ;                       STEPC   2=Byte, 6=Word, 14=Long
                                        ;9537  ;
                                        ;9538  ;       Outputs         D       Uncorrected remainder
                                        ;9539  ;                       Q       Quotient(magnitude)
                                        ;9540  ;                       FLAG3   Divisor positive
                                        ;9541  ;                       FLAG0   Quotient will be neg(when +- or -+)
                                        ;9542  ;                       PLATCH  Set to 100000 is Dividend is Negative
                                        ;9543  ;                       ALUS    Carry From Last Iteration For Rem Correction
                                        ;9544  ;
                                        ;9545  ;       Returns         -2      Quotient is pos
                                        ;9546  ;                       -1      Quotient is neg
                                        ;9547  ;*************************************************************************
                                        ;9548  =000
                                        ;9549  IL.DIVSUB:
                                        ;9550  IL.SOR+.END+:
                                        ;9551          ;000-----------------------------; DIVSUB SOR+ END+
                                        ;9552          DIVFAST+ SOR IN R[TEMP3],       ; DO SET UP CYCLE, SAVE ALU BORROW
                                        ;9553          SIZE[IDEP],ALUS_UNSGN,FLAG2?,   ; & BUT EDIV FLAG
U 0448, 0880,0026,C4B0,D0C7,0047,0      ;9554          NEXT/IL.DIVSUB.A                ;
                                        ;9555
                                        ;9556  IL.SOR+.END-:
                                        ;9557          ;001-----------------------------; DIVSUB SOR+ END-
                                        ;9558          Q_M[TEMP0]-Q,SET FLAG0,PL_[32.],; COMP Q(TEMP0=0), SET QUO NEG FLAG,
U 0449, 0140,0EC8,14B1,0047,0044,8      ;9559          F[AG2?,NEXT/IL.SOR+.END+        ; SET PL<4:0>=0 & PL<5>=1, EDIV?
                                        ;9560
                                        ;9561  IL.SOR-.END+:
                                        ;9562          ;010-----------------------------; DIVSUB SOR- END+
                                        ;9563          DIVFAST- SOR IN R[TEMP3],       ; DO SET UP CYCLE, SAVE ALU BORROW
                                        ;9564          ALUS_UNSGN,SIZE[IDEP],SET FLAG0,; SET QUO NEG FLAG,
U 044A, 0440,0027,C4B0,D0C7,0047,2      ;9565          FLAG2?,NEXT/IL.DIVSUB.B         ; & BUT EDIV FLAG
                                        ;9566
                                        ;9567  IL.SOR-.END-:
                                        ;9568          ;011-----------------------------; DIVSUB SOR- END-
                                        ;9569          Q_M[TEMP0]-Q,FLAG2?,PL_[32.],   ; COMPLEMENT Q(TEMP0=0),
U 044B, 0180,0EC8,14B1,0047,0044,2      ;9570          NEXT/IL.DIVSUB.C                ; SET PL<4:0>=0 & PL<5>=1
                                        ;9571
                                        ;9572          ;100-----------------------------; SOR+ END- EDIV
                                        ;9573          D_R[ZERO]-D-ALKC,               ; COMPLEMENT D
U 044C, 0C80,0063,203D,8047,0044,8      ;9574          NEXT/IL.SOR+.END+               ;
                                        ;9575  =
```

```
                                    ;9576  =0**
                                    ;9577  IL.DIVSUB.C:
                                    ;9578          ;0**-------------------------------;
                                    ;9579          DIVFAST- SOR IN R[TEMP3],         ; DO SET UP CYCLE, SAVE ALU BORROW
                                    ;9580          ALUS_UNSGN,SIZE[IDEP],FLAG2?,     ; & BUT EDIV FLAG
U 0442, 0480,0027,C4B0,D0C7,0047,2  ;9581          NEXT7IL.DIVSUB.B                  ;
                                    ;9582
                                    ;9583          ;**1-------------------------------;
                                    ;9584          D_R[ZERO]-D-ALKC,                 ; COMPLEMENT D
U 0446, 0C80,0063,203D,8047,0044,2  ;9585          NEXT/IL.DIVSUB.C                  ;
                                    ;9586  =0*0
                                    ;9587  IL.DIVSUB.A:
                                    ;9588          ;0*0-------------------------------; FLAG2 = 0 OR EDIV NOT V
                                    ;9589          DIVFAST+ SOR IN R[TEMP3],         ; STAY IN LOOP TILL STEPC = 0
                                    ;9590          SIZE[IDEP],                       ;
U 0470, 0480,0026,C330,C047,0047,0  ;9591          DBZ STEPC?,NEXT/IL.DIVSUB.A       ;
                                    ;9592
                                    ;9593          ;0*1-------------------------------;
                                    ;9594          DIVFAST+ SOR IN R[TEMP3],         ; STEPC = 0
                                    ;9595          SIZE[IDEP],                       ;
U 0471, 0080,0026,C430,C047,004B,0  ;9596          FLAG0?,NEXT/IL.DIVSUB.A1          ; WILL QUO BE POS OR NEG?
                                    ;9597
                                    ;9598          ;1*0-------------------------------; FLAG2 = 1 : EDIV
                                    ;9599          DIVFAST+ SOR IN R[TEMP3],         ; DO ITERATION
U 0474, 0498,0026,CB70,D847,0046,D  ;9600          SIZE[IDEP],DEC STEPC,ALUS?        ; DEC STEPC & CHECK FOR V
                                    ;9601  =
                                    ;9602  =01     ;01------------------------------; NO BORROW : EDIV V
U 046D, 0C80,0E77,10B0,0047,03FF,E  ;9603          Q_-1,RETURN [-2]                  ; Q <- -1 & TAKE QUO IS POS RETURN
                                    ;9604
                                    ;9605          ;11------------------------------; BORROW    : EDIV NO V
                                    ;9606          DIVFAST+ SOR IN R[TEMP3],         ;
                                    ;9607          SIZE[IDEP],                       ;
U 046F, 0498,0026,C030,C047,0047,0  ;9608          DEC STEPC,NEXT/IL.DIVSUB.A        ;
                                    ;9609  =0
                                    ;9610  IL.DIVSUB.A1:
                                    ;9611          ;0-------------------------------; QUO WIL BE POS,
                                    ;9612          DIVFAST+ SOR IN R[TEMP3],         ; DO LAST DIVFAST,
                                    ;9613          SET FLAG3,ALUS_UNSGN,SIZE[IDEP],; SAVE CARRY FOR REM CORRECTION,
U 04B0, 0458,0026,C0B0,D0C7,03FF,E  ;9614          RETURN [-2]                       ; SET DIVISOR POS FLAG
                                    ;9615
                                    ;9616          ;1-------------------------------; QUO WIL BE NEG,
                                    ;9617          DIVFAST+ SOR IN R[TEMP3],         ; DO LAST DIVFAST,
                                    ;9618          SET FLAG3,ALUS_UNSGN,SIZE[IDEP],; SAVE CARRY FOR REM CORRECTION,
U 04B1, 0C58,0026,C0B0,D0C7,03FF,F  ;9619          RETURN [-1]                       ; SET DIVISOR POS FLAG
```

```
                                    ;9620  =0*0
                                    ;9621  IL.DIVSUB.B:
                                    ;9622          ;0*0-------------------------------; FLAG2 = 0 OR EDIV NOT V
                                    ;9623          DIVFAST- SOR IN R[TEMP3],        ; STAY IN LOOP TILL STEPC = 0
                                    ;9624          SIZE[IDEP],                      ;
U 0472, 0880,0027,C330,C047,0047,2  ;9625          DBZ STEPC?,NEXT/IL.DIVSUB.B      ;
                                    ;9626
                                    ;9627          ;0*1-------------------------------;
                                    ;9628          DIVFAST- SOR IN R[TEMP3],        ; STEPC = 0
                                    ;9629          SIZE[IDEP],                      ;
U 0473, 0C80,0027,C430,C047,004B,2  ;9630          FLAG0?,NEXT/IL.DIVSUB.B1         ; WILL QUO BE POS OR NEG?
                                    ;9631
                                    ;9632          ;1*0-------------------------------; FLAG2 = 1 : EDIV
                                    ;9633          DIVFAST- SOR IN R[TEMP3],        ; DO ITERATION
U 0476, 0098,0027,CB70,D847,0047,5  ;9634          SIZE[IDEP],DEC STEPC,ALUS?       ; DEC STEPC & CHECK FOR V
                                    ;9635          =
                                    ;9636  =01     ;01------------------------------; NO BORROW : EDIV V
U 0475, 0C80,0E77,10B0,0047,03FF,E  ;9637          Q_-1,RETURN [-2]                 ; Q <- -1 & TAKE QUO IS POS RETURN
                                    ;9638
                                    ;9639          ;11------------------------------; BORROW    : EDIV NO V
                                    ;9640          DIVFAST- SOR IN R[TEMP3],        ;
                                    ;9641          SIZE[IDEP],                      ;
U 0477, 0898,0027,C030,C047,0047,2  ;9642          DEC STEPC,NEXT/IL.DIVSUB.B       ;
                                    ;9643  =0
                                    ;9644  IL.DIVSUB.B1:
                                    ;9645          ;0-------------------------------; QUO WIL BE POS,
                                    ;9646          DIVFAST- SOR IN R[TEMP3],        ; DO LAST DIVFAST,SAVE CARRY FOR REM
                                    ;9647          SIZE[IDEP],ALUS_UNSGN,           ; CORRECTION,SET DIVISOR POS FLAG
U 04B2, 0080,0027,C0B0,D0C7,03FF,E  ;9648          RETURN [-2]                      ;
                                    ;9649
                                    ;9650          ;1-------------------------------; QUO WIL BE NEG,
                                    ;9651          DIVFAST- SOR IN R[TEMP3],        ; DO LAST DIVFAST,SAVE CARRY FOR REM
                                    ;9652          ALUS_UNSGN,SIZE[IDEP],           ; CORRECTION,SET DIVISOR POS FLAG
U 04B3, 0880,0027,C0B0,D0C7,03FF,F  ;9653          RETURN [-1]                      ;
```

```
                                        ;9654  .TOC     ""      Integer, Logical, & Address     : ASH''
                                        ;9655
                                        ;9656  ;*****************************************************************************
                                        ;9657  ;       ASH(L,Q) cnt.rb src.rx dst.wx
                                        ;9658  ;          Input                                Q              Count
                                        ;9659  ;                                               MDR            Source
                                        ;9660  ;*****************************************************************************
                                        ;9661
                                        ;9662  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;9663  =0
                                        ;9664  IL.ASH: ;0-----------------------------; 78 79
                                        ;9665          PUSH,R[TEMP3] PL_Q Q_M[MDR],    ; TEMP3 & PLATCH <- CNT, Q <_ SOURCE
U 0092, 0C85,2BCC,71B0,C047,0414,0      ;9666          LOD INC BRA?,NEXT/OS.WRT1       ; GO EVALUATE DST
                                        ;9667
                                        ;9668          ;1-----------------------------;
                                        ;9669          M[TEMP2]_Q Q_D,SL_[20],MDR_0,   ; MTEMP2 <- SRC. SL <- 32. FOR ASHQ
U 0093, 0186,2F6C,7671,04E7,0043,E      ;9670          IR<2-0>?                        ; MDR <- 0 FOR ASHL(DON'T NEED Q_D)
                                        ;9671
                                        ;9672  .REGION/INTLOG.R1L,INTLOG.R1H/INTLOG.R2L,INTLOG.R2H/INTLOG.R3L,INTLOG.R3H
                                        ;9673  =110
                                        ;9674  IL.ASHL:
                                        ;9675          ;110---------------------------; 78
                                        ;9676          ABSVAL M[TEMP3]<7-0>?,          ; CHECK RANGE OF CNT
U 043E, 0480,3E52,0DF0,0047,0847,8 346* ;9677          NEXT/IL.ASHL.A                  ;
                                        ;9678
                                        ;9679  IL.ASHQ:
                                        ;9680          ;111---------------------------; 79
                                        ;9681          ABSVAL M[TEMP3]<7-0>?,          ; CHECK RANGE OF CNT
U 043F, 0C80,3E52,0DF0,0047,0848,0 346* ;9682          NEXT/IL.ASHQ.A                  ;
                                        ;9683  =00
                                        ;9684  IL.ASHL.A:
                                        ;9685          ;00----------------------------; CNT = -1 TO -31
                                        ;9686          R[DST.R] M[TEMP2].ASR.-P,CCOP1, ; DEST <- RESULT
U 0478, 0C84,2AB7,013C,4DDA,003F,9      ;9687          WRITE NOTREG,SIZE[IDEP],IRD1    ;
                                        ;9688
                                        ;9689          ;01----------------------------; CNT < -31
U 0479, 0180,0EF6,4031,0847,0047,8      ;9690          PL_[21],NEXT/IL.ASHL.A          ; SET PL TO -31 & GO TO SHIFT RIGHT
                                        ;9691
                                        ;9692          ;10----------------------------; CNT = 0 TO 31
                                        ;9693          Q_M[TEMP2].ASL.P,SIZE[IDEP],    ; Q <- RESULT
U 047A, 0880,2A77,1670,0047,0847,C 355* ;9694          SIGND CMP?,NEXT/IL.ASHL.B       ; LOOK AT SIGN OF RESULT
                                        ;9695
                                        ;9696          ;11----------------------------; CNT > 31
                                        ;9697          Q_M[MDR] WB_R[TEMP2],           ; RESULT <- 0
U 047B, 0481,2004,7A30,8047,004B,6      ;9698          WR.EQ.0?,NEXT/IL.ASHLWRTQ       ; IF SOURCE 0 THEN NO V
```

```
                                        ;9699  =00
                                        ;9700  IL.ASHL.B:
                                        ;9701          ;00-------------------------------; RESULT > 0
                                        ;9702          WB_M[TEMP2].ASR.-P,               ; WB <- SRC<31>'BITS SHIFTED OUT
U 047C, 0C30,2AB7,0A30,0047,084B,6 325* ;9703          WX.EQ.0?,NEXT/IL.ASHLWRTQ         ; IF 0 THEN NO V
                                        ;9704
                                        ;9705          ;01-------------------------------; RESULT = 0 (SRC=0 CNT=0 OR V)
                                        ;9706          ZEXT(M[TEMP3]).EQ.0?,SIZE[BYTE],  ; IF CNT 0 THEN NO V
U 047D, 0880,3016,4A0D,8047,004B,4      ;9707          NEXT/IL.ASHL.C                    ;
                                        ;9708
                                        ;9709          ;10-------------------------------; RESULT < 0
                                        ;9710          WB_NOT(M[TEMP2].ASR.-P),          ; WB <- NOT(SRC<31>'BITS SHIFTED OUT)
U 047E, 0880,2AB6,0A30,0047,084B,6 325* ;9711          WX.EQ.0?,NEXT/IL.ASHLWRTQ         ; IF 0 THEN NO V
                                        ;9712  =
                                        ;9713  =0
                                        ;9714  IL.ASHL.C:
                                        ;9715          ;0--------------------------------; CNT NE 0 (SRC=0 OR V)
                                        ;9716          WB_M[TEMP2].ASR.-P,               ; WB <- SRC<31>'BITS SHIFTED OUT
U 04B4, 0C80,2AB7,0A30,0047,084B,6 325* ;9717          WX.EQ.0?,NEXT/IL.ASHLWRTQ         ; IF 0 THEN NO V
                                        ;9718
                                        ;9719          ;1--------------------------------; CNT = 0
                                        ;9720          R[DST.R]_M[TEMP2],SIZE[IDEP],     ; WRITE SOURCE
U 04B5, 0C84,2592,413C 4DDA,003F,9      ;9721          WRITE NOTREG,CCOP1,IRD1           ;
                                        ;9722  =0
                                        ;9723  IL.ASHLWRTQ:
                                        ;9724          ;0--------------------------------; WRITE DEST SET V
                                        ;9725          R[DST.R]_Q_Q_D,WRITE NOTREG,      ; (DON'T CARE ABOUT Q_D)
                                        ;9726          SIZE[IDEP],CCOP1,SET MM.NOINT,    ;
U 04B6, 0C64,002C,703C,4DDA,0044,7      ;9727          NEXT/IL.MULSETV                   ;
                                        ;9728
                                        ;9729          ;1--------------------------------; WRTTE DEST NO V
                                        ;9730          R[DST.R]_Q_Q_D,WRITE NOTREG,      ; (DON'T CARE ABOUT Q_D)
U 04B7, 0884,002C,713C,4DDA,003F,9      ;9731          SIZE[IDEP],CCOP1,IRD1             ;
```

```
                                ;9732  =00
                                ;9733  IL.ASHQ.A:
                                ;9734          ;00-------------------------------; CNT = -1TO-31
                                ;9735          D (R[TEMP1] M[TEMP2]).RL.P,      ; D <- LS WORD OF RESULT, IF MEMORY
                                ;9736          WRITE NOTREG,CCOP1,SET MM.NOINT,; MODE THEN WRITE.
U 0480, 0060,28F7,2030,4DDA,0047,F  ;9737          SIZE[IDEP],NEXT/IL.ASHQ.A1      ;
                                ;9738
                                ;9739  IL.ASHQ.B:
                                ;9740          ;01-------------------------------; CNT < -31
                                ;9741          R[DST.R] M[TEMP2].ASR.-F,        ; WRITE <- LS WORD OF RESULT
                                ;9742          WRITE NOTREG,CCOP1,SET MM.NOINT,; (ASSUME CNT IS -32 TO -63 & FIX
U 0481, 0864,2AB7,003C,4DDA,0048,8  ;9743          SIZE[IDEP],NEXT/IL.ASHQ.B1      ; LATER IF NOT)
                                ;9744
                                ;9745  IL.ASHQ.C:
                                ;9746          ;10-------------------------------; CNT = 0-031
                                ;9747          R[DST.R] M[TEMP1].ASL.P,         ; WRITE LS WORD OF RESULT
                                ;9748          SIZE[IDEP],CCOP1,                ;
                                ;9749          WRITE NOTREG,SET MM.NOINT,       ;
U 0482, 0464,1A77,0DFC,4DDA,0048,9  ;9750          PL<4-0>.EQ.0?,NEXT/IL.ASHQ.C1   ; IS CNT = 0?
                                ;9751
                                ;9752  IL.ASHQ.D:
                                ;9753          ;11-------------------------------; CNT > 31
                                ;9754          R[DST.R] Q_0,WRITE NOTREG,       ; WRITE LS WORD OF RESULT
                                ;9755          SIZE[IDEP],CCOP1,SET MM.NOINT,   ;
U 0483, 0064,05B7,103C,4DDA,0048,A  ;9756          NEXT/IL.ASHQ.D1                 ;
                                ;9757
                                ;9758  IL.ASHQ.A1:
                                ;9759          ;-------------------------------; CNT = -1TO-31
U 047F, 0484,05B2,403C,4447,0048,4  ;9760          R[DST.R]_D,VA_VA+4              ; WRITE LS WORD IF REG MODE & BUMP VA
                                ;9761
                                ;9762          ;-------------------------------;
                                ;9763          R[DST.R+1] M[TEMP2].ASR.-P,      ; WRITE MS WORD OF RESULT
                                ;9764          CCOP2,WRITE NOTREG,SIZE[LONG],   ;
U 0484, 0884,2AB7,012F,55DA,003F,9  ;9765          IRD1                            ;
                                ;9766
                                ;9767  IL.ASHQ.B1:
                                ;9768          ;-------------------------------; CNT < -31
                                ;9769          VA_VA+4,                        ; BUMP VA
U 0488, 0580,3ED1,0DC1,FC47,0848,5 410*  ;9770          WB_M[TEMP3]+63 PL_31 WB<7>EQ0?  ; PL <- 31. IS CNT < -63?
                                ;9771  =00
                                ;9772  =01      ;01-------------------------------; CNT < -63
                                ;9773          R[DST.R] M[TEMP2].ASR.P,CCOP1,   ; WRITE LS WORD OF RESULT (IF REG)
                                ;9774          WRITE NOTREG,SIZE[LONG],         ; WRITE MS WORD OF RESULT (IF MEM)
U 0485, 0884,20F7,092C,4DDA,0048,6  ;9775          REG MODE?                       ;
                                ;9776
                                ;9777          ;10-------------------------------;
U 0486, 0D81,BC10,0030,24A7,0048,7  ;9778          VA_M[VA]-2LITO[4]               ; SET UP VA TO WRITE LS WORD MEM
                                ;9779
                                ;9780          ;11-------------------------------; CNT -32TO-63 (OR SEC WRT < -63)
                                ;9781          R[DST.R+1] M[TEMP2].ASR.P,CCOP2,; WRITE MS WORD OF RESULT (SRC<31>
U 0487, 0C84,20F7,012F,55DA,003F,9  ;9782          WRITE NOTREG,SIZE[LONG],IRD1    ;
```

```
                                        ;9783  =01
                                        ;9784  IL.ASHQ.C1:
                                        ;9785          ;01-------------------------------; CNT = 1TO31
                                        ;9786          Q_(R[TEMP1] M[TEMP2]).RL.P,      ; Q <- MS WORD OF RESULT
                                        ;9787          VA_VA+4,WB<31-30>?,              ; BUMP VA, & BRANCH ON SIGN
U 0489, 0480,28F7,16F0,4447,0048,D      ;9788          NEXT/IL.ASHQ.C2                  ;
                                        ;9789
                                        ;9790          ;11-------------------------------; CNT = 0
U 048B, 0080,2592,5030,0467,001E,5      ;9791          MDR_Q_M[TEMP2],NEXT/IL.MOVQ      ; SET UP TO DO MOVQ
                                        ;9792  =01
                                        ;9793  IL.ASHQ.C2:
                                        ;9794          ;01-------------------------------; POS
                                        ;9795          WB_M[TEMP2].ASR.-P,              ; WB <- SRC<63>'BITS SHIFTED OUT
U 048D, 0080,2AB7,0A70,0047,0049,8      ;9796          WX.NE.0?,NEXT/IL.ASHQ.WRT.MSW    ; IF 0 THEN NO V
                                        ;9797
                                        ;9798          ;11-------------------------------; NEG
                                        ;9799          WB_NOT(M[TEMP2].ASR.-P),         ; WB <- NOT(SRC<63>'BITS SHIFTED OUT)
U 048F, 0480,2AB6,0A70,0047,0049,8      ;9800          WX.NE.0?,NEXT/IL.ASHQ.WRT.MSW    ; IF 0 THEN NO V
                                        ;9801
                                        ;9802  IL.ASHQ.D1:
                                        ;9803          ;-------------------------------; CNT > 31
                                        ;9804          VA_VA+4,                         ; BUMP VA
U 048A, 0880,3B90,0DF0,0447,0849,0 413* ;9805          (M[TEMP3]-SL)BYTE RANGE CHECK?   ; IS CNT (=32, 33TO 63, >63)?
                                        ;9806
                                        ;9807  =00     ;00-------------------------------; CNT 33TO63
                                        ;9808          Q_M[TEMP1].ASL.P,                ; Q <- MS WORD OF RESULT
U 0490, 0880 1A77,16F0,0047,0049,5      ;9809          WB<31-30>?,NEXT/IL.ASHQ.D2       ; BRANCH ON SIGN
                                        ;9810
                                        ;9811          ;01-------------------------------; CNT =32
                                        ;9812          Q_M[TEMP1],                      ; Q <- MS WORD OF RESULT
U 0491, 0880,1592,56F0,0047,0048,D      ;9813          WB<31-30>?,NEXT/IL.ASHQ.C2       ; GO CHECK FOR V
                                        ;9814
                                        ;9815          ;10-------------------------------; CNT =64
                                        ;9816          WB_M[TEMP1].OR.R[TEMP2],         ; V IF SRC NOT = 0
U 0492, 0480,1002,4A70,8047,0049,8      ;9817          WX.NE.0?,NEXT/IL.ASHQ.WRT.MSW    ;
                                        ;9818
                                        ;9819          ;11-------------------------------; CNT >64
                                        ;9820          WB_M[TEMP1].OR.R[TEMP2],         ; V IF SRC NOT = 0
U 0493, 0480,1002,4A70,8047,0049,8      ;9821          WX.NE.0?,NEXT/IL.ASHQ.WRT.MSW    ; IF 0 THEN NO V
                                        ;9822  =01
                                        ;9823  IL.ASHQ.D2:
                                        ;9824          ;0--------------------------------; RESULT POS
                                        ;9825          WB_R[TEMP2].OR.(M[TEMP1].ASR.-P),; WB <- BITS SHIFTED OUT
U 0495, 0880,1ABE,4A70,8047,0849,8 321* ;9826          WX.NE.0?,NEXT/IL.ASHQ.WRT.MSW    ; IF 0 THEN NO V
                                        ;9827
                                        ;9828          ;1--------------------------------; RESULT NEG
                                        ;9829          M[TEMP2].EQ.-1?,SIZE[IDEP],      ; IF = -1 CHK TEMP1 FOR V
U 0497, 0080,2F53,CDB0,0047,0849,9 346* ;9830          NEXT/IL.ASHQ.WRT.MSW.SETV        ; IF NOT = -1 THEN V
```

```
                                        ;9831  =00
                                        ;9832  IL.ASHQ.WRT.MSW:
                                        ;9833          ;00-----------------------------; WRITE DEST NO V
                                        ;9834          R[DST.R+1]_Q Q_D,WRITE NOTREG,  ; (DON'T CARE ABOUT Q_D)
U 0498, 0C84,002C,712F,55DA,003F,9      ;9835          CCOP2,SIZE[LONG],IRD1           ;
                                        ;9836
                                        ;9837  IL.ASHQ.WRT.MSW.SETV:
                                        ;9838          ;01-----------------------------; WRITE DEST SET V
                                        ;9839          R[DST.R+1]_Q Q_D,WRITE NOTREG,  ; (DON'T CARE ABOUT Q_D)
U 0499, 0084,002C,702F,55DA,0044,7      ;9840          CCOP2,SIZE[LONG],NEXT/IL.MULSETV;
                                        ;9841
                                        ;9842  =11     ;11-----------------------------; MTEMP2 = -1
                                        ;9843          WB_NOT(M[TEMP1].ASR.-P),        ; WB <- NOT(BITS SHIFTED OUT)
U 049B, 0480,1AB6,0A70,0047,0049,8      ;9844          WX.NE.0?,NEXT/IL.ASHQ.WRT.MSW   ; IF 0 THEN NO V
```

```
                                 .TOC   "     Integer, Logical, & Address     : Ird Rom Definition"
 .NOBIN

 .ICODE  ;      OPS            REG                           MEM              OPS          FPA REG                 FPA MEM
        58: FPD [NOP][IE.OPCOD.DEC              ]                             [NOP][IE.OPCOD.DEC         ],                       ;ADAWI
            IRD1[LOD][OS.RED                    ]                             [LOD][OS.RED               ]
 .OCODE
        58: CNTO[LOD][IL.ADAWIREG         ][OS.WRT2                ]  [LOD][IL.ADAWIREG        ][OS.WRT2                ],
            CNT1[NOP][IL.ADAWIMEM         ][IL.ADAWIMEM            ]  [NOP][IL.ADAWIMEM        ][IL.ADAWIMEM            ]

 .ICODE
        80: FPD [NOP][IE.OPCOD.DEC              ]                             [NOP][IE.OPCOD.DEC         ],                       ;ADDB2
            IRD1[LOD][OS.RED                    ]                             [LOD][OS.RED               ]
 .OCODE
        80: CNTO[LOD][IL.ADD2.B.W.L.REG   ][OS.MOD                 ]  [LOD][IL.ADD2.B.W.L.REG  ][OS.MOD                 ],
            CNT1[NOP][IL.ADD2.B.W.L.MEM   ][IL.ADD2.B.W.L.MEM      ]  [NOP][IL.ADD2.B.W.L.MEM  ][IL.ADD2.B.W.L.MEM      ]

 .ICODE
        81: FPD [NOP][IE.OPCOD.DEC              ]                             [NOP][IE.OPCOD.DEC         ],                       ;ADDB3
            IRD1[LOD][OS.RED                    ]                             [LOD][OS.RED               ]
 .OCODE
        81: CNTO[LOD][OS.RED              ][OS.RED                 ]  [LOD][OS.RED             ][OS.RED                 ],
            CNT1[LOD][IL.ADD3.B.W.L.REG   ][IL.ADD3.B.W.L.MEM      ]  [LOD][IL.ADD3.B.W.L.REG  ][IL.ADD3.B.W.L.MEM      ]

 .ICODE
       0C0: FPD [NOP][IE.OPCOD.DEC              ]                             [NOP][IE.OPCOD.DEC         ],                       ;ADDL2
            IRD1[LOD][OS.RED                    ]                             [LOD][OS.RED               ]
 .OCODE
       0C0: CNTO[LOD][IL.ADD2.B.W.L.REG   ][OS.MOD                 ]  [LOD][IL.ADD2.B.W.L.REG  ][OS.MOD                 ],
            CNT1[NOP][IL.ADD2.B.W.L.MEM   ][IL.ADD2.B.W.L.MEM      ]  [NOP][IL.ADD2.B.W.L.MEM  ][IL.ADD2.B.W.L.MEM      ]

 .ICODE
       0C1: FPD [NOP][IE.OPCOD.DEC              ]                             [NOP][IE.OPCOD.DEC         ],                       ;ADDL3
            IRD1[LOD][OS.RED                    ]                             [LOD][OS.RED               ]
 .OCODE
       0C1: CNTO[LOD][OS.RED              ][OS.RED                 ]  [LOD][OS.RED             ][OS.RED                 ],
            CNT1[LOD][IL.ADD3.B.W.L.REG   ][IL.ADD3.B.W.L.MEM      ]  [LOD][IL.ADD3.B.W.L.REG  ][IL.ADD3.B.W.L.MEM      ]

 .ICODE
       0A0: FPD [NOP][IE.OPCOD.DEC              ]                             [NOP][IE.OPCOD.DEC         ],                       ;ADDW2
            IRD1[LOD][OS.RED                    ]                             [LOD][OS.RED               ]
 .OCODE
       0A0: CNTO[LOD][IL.ADD2.B.W.L.REG   ][OS.MOD                 ]  [LOD][IL.ADD2.B.W.L.REG  ][OS.MOD                 ],
            CNT1[NOP][IL.ADD2.B.W.L.MEM   ][IL.ADD2.B.W.L.MEM      ]  [NOP][IL.ADD2.B.W.L.MEM  ][IL.ADD2.B.W.L.MEM      ]

 .ICODE
       0A1: FPD [NOP][IE.OPCOD.DEC              ]                             [NOP][IE.OPCOD.DEC         ],                       ;ADDW3
            IRD1[LOD][OS.RED                    ]                             [LOD][OS.RED               ]
 .OCODE
       0A1: CNTO[LOD][OS.RED              ][OS.RED                 ]  [LOD][OS.RED             ][OS.RED                 ],
            CNT1[LOD][IL.ADD3.B.W.L.REG   ][IL.ADD3.B.W.L.MEM      ]  [LOD][IL.ADD3.B.W.L.REG  ][IL.ADD3.B.W.L.MEM      ]
```

```
 .ICODE  ;      OPS              REG                      MEM              OPS           FPA REG               FPA MEM
       0D8: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC         ],                           ;ADWC
            IRD1[LOD][OS.RED              ]                          [LOD][OS.RED               ]
 .OCODE
       0D8: CNT0[LOD][IL.ADWCREG          ][OS.MOD                 ] [LOD][IL ADWCREG           ][OS.MOD               ],
            CNT1[NOP][IL.ADWCMEM          ][IL.ADWCMEM             ] [NOP][IL.ADWCMEM           ][IL.ADWCMEM           ]

 .ICODE
        78: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC         ],                           ;ASHL
            IRD1[LOD][OS.RED              ]                          [LOD][OS.RED               ]
 .OCODE
        78: CNT0[LOD][OS.RED              ][OS.RED                 ] [LOD][OS.RED               ][OS.RED               ],
            CNT1[NOP][IL.ASH              ][IL.ASH                 ] [NOP][IL.ASH               ][IL.ASH               ]

 .ICODE
        79: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC         ],                           ;ASHQ
            IRD1[LOD][OS.RED              ]                          [LOD][OS.RED               ]
 .OCODE
        79: CNT0[LOD][OS.QRED             ][OS.QRED                ] [LOD][OS.QRED              ][OS.QRED              ],
            CNT1[NOP][IL.ASH              ][IL.ASH                 ] [NOP][IL.ASH               ][IL.ASH               ]

 .ICODE
        8A: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC         ],                           ;BICB2
            IRD1[LOD][OS.RED              ]                          [LOD][OS.RED               ]
 .OCODE
        8A: CNT0[LOD][IL.BIC2.B.W.L.REG   ][OS.MOD                 ] [LOD][IL.BIC2.B.W.L.REG    ][OS.MOD               ],
            CNT1[NOP][IL.BIC2.B.W.L.MEM   ][IL.BIC2.B.W.L.MEM      ] [NOP][IL.BIC2.B.W.L.MEM    ][IL.BIC2.B.W.L.MEM    ]

 .ICODE
        8B: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC         ],                           ;BICB3
            IRD1[LOD][OS.RED              ]                          [LOD][OS.RED               ]
 .OCODE
        8B: CNT0[LOD][OS.RED              ][OS.RED                 ] [LOD][OS.RED               ][OS.RED               ],
            CNT1[LOD][IL.BIC3.B.W.L.REG   ][IL.BIC3.B.W.L.MEM      ] [LOD][IL.BIC3.B.W.L.REG    ][IL.BIC3.B.W.L.MEM    ]

 .ICODE
       0CA: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC         ],                           ;BICL2
            IRD1[LOD][OS.RED              ]                          [LOD][OS.RED               ]
 .OCODE
       0CA: CNT0[LOD][IL.BIC2.B.W.L.REG   ][OS.MOD                 ] [LOD][IL.BIC2.B.W.L.REG    ][OS.MOD               ],
            CNT1[NOP][IL.BIC2.B.W.L.MEM   ][IL.BIC2.B.W.L.MEM      ] [NOP][IL.BIC2.B.W.L.MEM    ][IL.BIC2.B.W.L.MEM    ]

 .ICODE
       0CB: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC         ],                           ;BICL3
            IRD1[LOD][OS.RED              ]                          [LOD][OS.RED               ]
 .OCODE
       0CB: CNT0[LOD][OS.RED              ][OS.RED                 ] [LOD][OS.RED               ][OS.RED               ],
            CNT1[LOD][IL.BIC3.B.W.L.REG   ][IL.BIC3.B.W.L.MEM      ] [LOD][IL.BIC3.B.W.L.REG    ][IL.BIC3.B.W.L.MEM    ]

 .ICODE
       0AA: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC         ],                           ;BICW2
            IRD1[LOD][OS.RED              ]                          [LOD][OS.RED               ]
 .OCODE
       0AA: CNT0[LOD][IL.BIC2.B.W.L.REG   ][OS.MOD                 ] [LOD][IL.BIC2.B.W.L.REG    ][OS.MOD               ],
            CNT1[NOP][IL.BIC2.B.W.L.MEM   ][IL.BIC2.B.W.L.MEM      ] [NOP][IL.BIC2.B.W.L.MEM    ][IL.BIC2.B.W.L.MEM    ]
```

```
 .ICODE  ;      OPS          REG                    MEM                 OPS       FPA REG              FPA MEM
        0AB: FPD [NOP][IE.OPCOD.DEC       ]                            [NOP][IE.OPCOD.DEC      ],                          ;BICW3
             IRD1[LOD][OS.RED             ]                            [LOD][OS.RED            ]
 .OCODE
        0AB: CNT0[LOD][OS.RED             ][OS.RED                ]    [LOD][OS.RED            ][OS.RED                ],
             CNT1[LOD][IL.BIC3.B.W.L.REG  ][IL.BIC3.B.W.L.MEM     ]    [LOD][IL.BIC3.B.W.L.REG ][IL.BIC3.B.W.L.MEM     ]

 .ICODE
         88: FPD [NOP][IE.OPCOD.DEC       ]                            [NOP][IE.OPCOD.DEC      ],                          ;BISB2
             IRD1[LOD][OS.RED             ]                            [LOD][OS.RED            ]
 .OCODE
         88: CNT0[LOD][IL.BIS2.B.W.L.REG  ][OS.MOD                ]    [LOD][IL.BIS2.B.W.L.REG ][OS.MOD                ],
             CNT1[NOP][IL.BIS2.B.W.L.MEM  ][IL.BIS2.B.W.L.MEM     ]    [NOP][IL.BIS2.B.W.L.MEM ][IL.BIS2.B.W.L.MEM     ]

 .ICODE
         89: FPD [NOP][IE.OPCOD.DEC       ]                            [NOP][IE.OPCOD.DEC      ],                          ;BISB3
             IRD1[LOD][OS.RED             ]                            [LOD][OS.RED            ]
 .OCODE
         89: CNT0[LOD][OS.RED             ][OS.RED                ]    [LOD][OS.RED            ][OS.RED                ],
             CNT1[LOD][IL.BIS3.B.W.L.REG  ][IL.BIS3.B.W.L.MEM     ]    [LOD][IL.BIS3.B.W.L.REG ][IL.BIS3.B.W.L.MEM     ]

 .ICODE
        0C8: FPD [NOP][IE.OPCOD.DEC       ]                            [NOP][IE.OPCOD.DEC      ],                          ;BISL2
             IRD1[LOD][OS.RED             ]                            [LOD][OS.RED            ]
 .OCODE
        0C8: CNT0[LOD][IL.BIS2.B.W.L.REG  ][OS.MOD                ]    [LOD][IL.BIS2.B.W.L.REG ][OS.MOD                ],
             CNT1[NOP][IL.BIS2.B.W.L.MEM  ][IL.BIS2.B.W.L.MEM     ]    [NOP][IL.BIS2.B.W.L.MEM ][IL.BIS2.B.W.L.MEM     ]

 .ICODE
        0C9: FPD [NOP][IE.OPCOD.DEC       ]                            [NOP][IE.OPCOD.DEC      ],                          ;BISL3
             IRD1[LOD][OS.RED             ]                            [LOD][OS.RED            ]
 .OCODE
        0C9: CNT0[LOD][OS.RED             ][OS.RED                ]    [LOD][OS.RED            ][OS.RED                ],
             CNT1[LOD][IL.BIS3.B.W.L.REG  ][IL.BIS3.B.W.L.MEM     ]    [LOD][IL.BIS3.B.W.L.REG ][IL.BIS3.B.W.L.MEM     ]

 .ICODE
        0A8: FPD [NOP][IE.OPCOD.DEC       ]                            [NOP][IE.OPCOD.DEC      ],                          ;BISW2
             IRD1[LOD][OS.RED             ]                            [LOD][OS.RED            ]
 .OCODE
        0A8: CNT0[LOD][IL.BIS2.B.W.L.REG  ][OS.MOD                ]    [LOD][IL.BIS2.B.W.L.REG ][OS.MOD                ],
             CNT1[NOP][IL.BIS2.B.W.L.MEM  ][IL.BIS2.B.W.L.MEM     ]    [NOP][IL.BIS2.B.W.L.MEM ][IL.BIS2.B.W.L.MEM     ]

 .ICODE
        0A9: FPD [NOP][IE.OPCOD.DEC       ]                            [NOP][IE.OPCOD.DEC      ],                          ;BISW3
             IRD1[LOD][OS.RED             ]                            [LOD][OS.RED            ]
 .OCODE
        0A9: CNT0[LOD][OS.RED             ][OS.RED                ]    [LOD][OS.RED            ][OS.RED                ],
             CNT1[LOD][IL.BIS3.B.W.L.REG  ][IL.BIS3.B.W.L.MEM     ]    [LOD][IL.BIS3.B.W.L.REG ][IL.BIS3.B.W.L.MEM     ]

 .ICODE
         93: FPD [NOP][IE.OPCOD.DEC       ]                            [NOP][IE.OPCOD.DEC      ],                          ;BITB
             IRD1[LOD][OS.RED             ]                            [LOD][OS.RED            ]
 .OCODE
         93: CNT0[LOD][IL.BIT.B.W.L.REG   ][OS.RED                ]    [LOD][IL.BIT.B.W.L.REG  ][OS.RED                ],
             CNT1[NOP][IL.BIT.B.W.L.MEM   ][IL.BIT.B.W.L.MEM      ]    [NOP][IL.BIT.B.W.L.MEM  ][IL.BIT.B.W.L.MEM      ]
```

```
;10006  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;10007       0D3: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                    ;BITL
;10008            IRD1[LOD][OS.RED              ]                        [LOD][OS.RED              ]
;10009  .OCODE
;10010       0D3: CNT0[LOD][IL.BIT.B.W.L.REG    ][OS.RED              ]  [LOD][IL.BIT.B.W.L.REG    ][OS.RED              ],
;10011            CNT1[NOP][IL.BIT.B.W.L.MEM    ][IL.BIT.B.W.L.MEM    ]  [NOP][IL.BIT.B.W.L.MEM    ][IL.BIT.B.W.L.MEM    ]
;10012
;10013  .ICODE
;10014       0B3: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                    ;BITW
;10015            IRD1[LOD][OS.RED              ]                        [LOD][OS.RED              ]
;10016  .OCODE
;10017       0B3: CNT0[LOD][IL.BIT.B.W.L.REG    ][OS.RED              ]  [LOD][IL.BIT.B.W.L.REG    ][OS.RED              ],
;10018            CNT1[NOP][IL.BIT.B.W.L.MEM    ][IL.BIT.B.W.L.MEM    ]  [NOP][IL.BIT.B.W.L.MEM    ][IL.BIT.B.W.L.MEM    ]
;10019
;10020  .ICODE
;10021       094: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                    ;CLRB
;10022            IRD1[LOD][OS.WRT1             ]                        [LOD][OS.WRT1             ]
;10023  .OCODE
;10024       094: CNT0[NOP][IL.CLR.B.W.L        ][IL.CLR.B.W.L        ]  [NOP][IL.CLR.B.W.L        ][IL.CLR.B.W.L        ],
;10025            CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
;10026
;10027  .ICODE
;10028       0D4: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                    ;CLRL
;10029            IRD1[LOD][OS.WRT1             ]                        [LOD][OS.WRT1             ]
;10030  .OCODE
;10031       0D4: CNT0[NOP][IL.CLR.B.W.L        ][IL.CLR.B.W.L        ]  [NOP][IL.CLR.B.W.L        ][IL.CLR.B.W.L        ],
;10032            CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
;10033
;10034  .ICODE
;10035       07C: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                    ;CLRQ
;10036            IRD1[LOD][OS.WRT1             ]                        [LOD][OS.WRT1             ]
;10037  .OCODE
;10038       07C: CNT0[NOP][IL.CLRQ             ][IL.CLRQ             ]  [NOP][IL.CLRQ             ][IL.CLRQ             ],
;10039            CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
;10040
;10041  .ICODE
;10042       0B4: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                    ;CLRW
;10043            IRD1[LOD][OS.WRT1             ]                        [LOD][OS.WRT1             ]
;10044  .OCODE
;10045       0B4: CNT0[NOP][IL.CLR.B.W.L        ][IL.CLR.B.W.L        ]  [NOP][IL.CLR.B.W.L        ][IL.CLR.B.W.L        ],
;10046            CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
;10047
;10048  .ICODE
;10049       091: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                    ;CMPB
;10050            IRD1[LOD][OS.RED              ]                        [LOD][OS.RED              ]
;10051  .OCODE
;10052       091: CNT0[LOD][IL.CMP.B.W.L.REG    ][OS.RED              ]  [LOD][IL.CMP.B.W.L.REG    ][OS.RED              ],
;10053            CNT1[NOP][IL.CMP.B.W.L.MEM    ][IL.CMP.B.W.L.MEM    ]  [NOP][IL.CMP.B.W.L.MEM    ][IL.CMP.B.W.L.MEM    ]
;10054
;10055  .ICODE
;10056       0D1: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                    ;CMPL
;10057            IRD1[LOD][OS.RED              ]                        [LOD][OS.RED              ]
;10058  .OCODE
;10059       0D1: CNT0[LOD][IL.CMP.B.W.L.REG    ][OS.RED              ]  [LOD][IL.CMP.B.W.L.REG    ][OS.RED              ],
;10060            CNT1[NOP][IL.CMP.B.W.L.MEM    ][IL.CMP.B.W.L.MEM    ]  [NOP][IL.CMP.B.W.L.MEM    ][IL.CMP.B.W.L.MEM    ]
```

```
;10061  .ICODE  ;       OPS             REG                     MEM             OPS         FPA REG                 FPA MEM
;10062          0B1: FPD [NOP][IE.OPCOD.DEC         ]                       [NOP][IE.OPCOD.DEC         ],                      ;CMPW
;10063               IRD1[LOD][OS.RED               ]                       [LOD][OS.RED               ]
;10064  .OCODE
;10065          0B1: CNT0[LOD][IL.CMP.B.W.L.REG     ][OS.RED                ] [LOD][IL.CMP.B.W.L.REG     ][OS.RED                ],
;10066               CNT1[NOP][IL.CMP.B.W.L.MEM     ][IL.CMP.B.W.L.MEM      ] [NOP][IL.CMP.B.W.L.MEM     ][IL.CMP.B.W.L.MEM      ]
;10067
;10068  .ICODE
;10069          098: FPD [NOP][IE.OPCOD.DEC         ]                       [NOP][IE.OPCOD.DEC         ],                      ;CVTBL
;10070               IRD1[LOD][OS.RED               ]                       [LOD][OS.RED               ]
;10071  .OCODE
;10072          098: CNT0[NOP][IL.CVT.BW.BL.WL      ][IL.CVT.BW.BL.WL       ] [NOP][IL.CVT.BW.BL.WL      ][IL.CVT.BW.BL.WL       ],
;10073               CNT1[NOP][IL.DIVQUO+           ][IL.DIVQUO+            ] [NOP][IL.DIVQUO+           ][IL.DIVQUO+            ]
;10074
;10075  .ICODE
;10076          099: FPD [NOP][IE.OPCOD.DEC         ]                       [NOP][IE.OPCOD.DEC         ],                      ;CVTBW
;10077               IRD1[LOD][OS.RED               ]                       [LOD][OS.RED               ]
;10078  .OCODE
;10079          099: CNT0[NOP][IL.CVT.BW.BL.WL      ][IL.CVT.BW.BL.WL       ] [NOP][IL.CVT.BW.BL.WL      ][IL.CVT.BW.BL.WL       ],
;10080               CNT1[NOP][IL.DIVQUO+           ][IL.DIVQUO+            ] [NOP][IL.DIVQUO+           ][IL.DIVQUO+            ]
;10081
;10082  .ICODE
;10083          0F6: FPD [NOP][IE.OPCOD.DEC         ]                       [NOP][IE.OPCOD.DEC         ],                      ;CVTLB
;10084               IRD1[LOD][OS.RED               ]                       [LOD][OS.RED               ]
;10085  .OCODE
;10086          0F6: CNT0[LOD][IL.CVT.LB.LW.WB.REG  ][OS.WRT2               ] [LOD][IL.CVT.LB.LW.WB.REG  ][OS.WRT2               ],
;10087               CNT1[NOP][IL.CVT.LB.LW.WB.MEM  ][IL.CVT.LB.LW.WB.MEM   ] [NOP][IL.CVT.LB.LW.WB.MEM  ][IL.CVT.LB.LW.WB.MEM   ]
;10088
;10089  .ICODE
;10090          0F7: FPD [NOP][IE.OPCOD.DEC         ]                       [NOP][IE.OPCOD.DEC         ],                      ;CVTLW
;10091               IRD1[LOD][OS.RED               ]                       [LOD][OS.RED               ]
;10092  .OCODE
;10093          0F7: CNT0[LOD][IL.CVT.LB.LW.WB.REG  ][OS.WRT2               ] [LOD][IL.CVT.LB.LW.WB.REG  ][OS.WRT2               ],
;10094               CNT1[NOP][IL.CVT.LB.LW.WB.MEM  ][IL.CVT.LB.LW.WB.MEM   ] [NOP][IL.CVT.LB.LW.WB.MEM  ][IL.CVT.LB.LW.WB.MEM   ]
;10095
;10096  .ICODE
;10097          033: FPD [NOP][IE.OPCOD.DEC         ]                       [NOP][IE.OPCOD.DEC         ],                      ;CVTWB
;10098               IRD1[LOD][OS.RED               ]                       [LOD][OS.RED               ]
;10099  .OCODE
;10100          033: CNT0[LOD][IL.CVT.LB.LW.WB.REG  ][OS.WRT2               ] [LOD][IL.CVT.LB.LW.WB.REG  ][OS.WRT2               ],
;10101               CNT1[NOP][IL.CVT.LB.LW.WB.MEM  ][IL.CVT.LB.LW.WB.MEM   ] [NOP][IL.CVT.LB.LW.WB.MEM  ][IL.CVT.LB.LW.WB.MEM   ]
;10102
;10103  .ICODE
;10104          032: FPD [NOP][IE.OPCOD.DEC         ]                       [NOP][IE.OPCOD.DEC         ],                      ;CVTWL
;10105               IRD1[LOD][OS.RED               ]                       [LOD][OS.RED               ]
;10106  .OCODE
;10107          032: CNT0[NOP][IL.CVT.BW.BL.WL      ][IL.CVT.BW.BL.WL       ] [NOP][IL.CVT.BW.BL.WL      ][IL.CVT.BW.BL.WL       ],
;10108               CNT1[NOP][IL.DIVQUO+           ][IL.DIVQUO+            ] [NOP][IL.DIVQUO+           ][IL.DIVQUO+            ]
;10109
;10110  .ICODE
;10111          097: FPD [NOP][IE.OPCOD.DEC         ]                       [NOP][IE.OPCOD.DEC         ],                      ;DECB
;10112               IRD1[LOD][OS.MOD               ]                       [LOD][OS.MOD               ]
;10113  .OCODE
;10114          097: CNT0[NOP][IL.DEC.B.W.L         ][IL.DEC.B.W.L          ] [NOP][IL.DEC.B.W.L         ][IL.DEC.B.W.L          ],
;10115               CNT1[NOP][IE.BAD.IRD           ][IE.BAD.IRD            ] [NOP][IE.BAD.IRD           ][IE.BAD.IRD            ]
```

```
;10116  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;10117       0D7: FPD [NOP][IE.OPCOD.DEC        ]                       [NOP][IE.OPCOD.DEC        ],                    ;DECL
;10118            IRD1[LOD][OS.MOD              ]                       [LOD][OS.MOD              ]
;10119  .OCODE
;10120       0D7: CNT0[NOP][IL.DEC.B.W.L        ][IL.DEC.B.W.L        ] [NOP][IL.DEC.B.W.L        ][IL.DEC.B.W.L        ],
;10121            CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ] [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
;10122
;10123  .ICODE
;10124       0B7: FPD [NOP][IE.OPCOD.DEC        ]                       [NOP][IE.OPCOD.DEC        ],                    ;DECW
;10125            IRD1[LOD][OS.MOD              ]                       [LOD][OS.MOD              ]
;10126  .OCODE
;10127       0B7: CNT0[NOP][IL.DEC.B.W.L        ][IL.DEC.B.W.L        ] [NOP][IL.DEC.B.W.L        ][IL.DEC.B.W.L        ],
;10128            CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ] [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
;10129
;10130  .ICODE
;10131       086: FPD [NOP][IE.OPCOD.DEC        ]                       [NOP][IE.OPCOD.DEC        ],                    ;DIVB2
;10132            IRD1[LOD][OS.RED              ]                       [LOD][OS.RED              ]
;10133  .OCODE
;10134       086: CNT0[NOP][IL.DIVB2            ][IL.DIVB2            ] [NOP][IL.DIVB2            ][IL.DIVB2            ],
;10135            CNT1[NOP][IL.DIV2.B.W.L       ][IL.DIV2.B.W.L       ] [NOP][IL.DIV2.B.W.L       ][IL.DIV2.B.W.L       ]
;10136
;10137  .ICODE
;10138       087: FPD [NOP][IE.OPCOD.DEC        ]                       [NOP][IE.OPCOD.DEC        ],                    ;DIVB3
;10139            IRD1[LOD][OS.RED              ]                       [LOD][OS.RED              ]
;10140  .OCODE
;10141       087: CNT0[NOP][IL.DIVB3            ][IL.DIVB3            ] [NOP][IL.DIVB3            ][IL.DIVB3            ],
;10142            CNT1[NOP][IL.DIV3.B.W.L       ][IL.DIV3.B.W.L       ] [NOP][IL.DIV3.B.W.L       ][IL.DIV3.B.W.L       ]
;10143
;10144  .ICODE
;10145       0C6: FPD [NOP][IE.OPCOD.DEC        ]                       [NOP][IE.OPCOD.DEC        ],                    ;DIVL2
;10146            IRD1[LOD][OS.RED              ]                       [LOD][OS.RED              ]
;10147  .OCODE
;10148       0C6: CNT0[NOP][IL.DIVL2            ][IL.DIVL2            ] [NOP][IL.DIVL2            ][IL.DIVL2            ],
;10149            CNT1[NOP][IL.DIV2.B.W.L       ][IL.DIV2.B.W.L       ] [NOP][IL.DIV2.B.W.L       ][IL.DIV2.B.W.L       ]
;10150
;10151  .ICODE
;10152       0C7: FPD [NOP][IE.OPCOD.DEC        ]                       [NOP][IE.OPCOD.DEC        ],                    ;DIVL3
;10153            IRD1[LOD][OS.RED              ]                       [LOD][OS.RED              ]
;10154  .OCODE
;10155       0C7: CNT0[NOP][IL.DIVL3            ][IL.DIVL3            ] [NOP][IL.DIVL3            ][IL.DIVL3            ],
;10156            CNT1[NOP][IL.DIV3.B.W.L       ][IL.DIV3.B.W.L       ] [NOP][IL.DIV3.B.W.L       ][IL.DIV3.B.W.L       ]
;10157
;10158  .ICODE
;10159       0A6: FPD [NOP][IE.OPCOD.DEC        ]                       [NOP][IE.OPCOD.DEC        ],                    ;DIVW2
;10160            IRD1[LOD][OS.RED              ]                       [LOD][OS.RED              ]
;10161  .OCODE
;10162       0A6: CNT0[NOP][IL.DIVW2            ][IL.DIVW2            ] [NOP][IL.DIVW2            ][IL.DIVW2            ],
;10163            CNT1[NOP][IL.DIV2.B.W.L       ][IL.DIV2.B.W.L       ] [NOP][IL.DIV2.B.W.L       ][IL.DIV2.B.W.L       ]
;10164
;10165  .ICODE
;10166       0A7: FPD [NOP][IE.OPCOD.DEC        ]                       [NOP][IE.OPCOD.DEC        ],                    ;DIVW3
;10167            IRD1[LOD][OS.RED              ]                       [LOD][OS.RED              ]
;10168  .OCODE
;10169       0A7: CNT0[NOP][IL.DIVW3            ][IL.DIVW3            ] [NOP][IL.DIVW3            ][IL.DIVW3            ],
;10170            CNT1[NOP][IL.DIV3.B.W.L       ][IL.DIV3.B.W.L       ] [NOP][IL.DIV3.B.W.L       ][IL.DIV3.B.W.L       ]
```

```
10171  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
10172       07B: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;EDIV
10173            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
10174  .OCODE
10175       07B: CNT0[LOD][OS.QRED           ][OS.QRED          ]  [LOD][OS.QRED           ][OS.QRED          ],
10176            CNT1[NOP][IL.EDIV           ][IL.EDIV          ]  [NOP][IL.EDIV           ][IL.EDIV          ]
10177
10178  .ICODE
10179       07A: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;EMUL
10180            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
10181  .OCODE
10182       07A: CNT0[LOD][OS.RED            ][OS.RED           ]  [LOD][OS.RED            ][OS.RED           ],
10183            CNT1[NOP][IL.EMUL           ][IL.EMUL          ]  [NOP][FI.EMUL           ][FI.EMUL          ]
10184
10185  .ICODE
10186       096: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;INCB
10187            IRD1[LOD][OS.MOD            ]                          [LOD][OS.MOD            ]
10188  .OCODE
10189       096: CNT0[NOP][IL.INC.B.W.L      ][IL.INC.B.W.L     ]  [NOP][IL.INC.B.W.L      ][IL.INC.B.W.L     ],
10190            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD       ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD       ]
10191
10192  .ICODE
10193       0D6: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;INCL
10194            IRD1[LOD][OS.MOD            ]                          [LOD][OS.MOD            ]
10195  .OCODE
10196       0D6: CNT0[NOP][IL.INC.B.W.L      ][IL.INC.B.W.L     ]  [NOP][IL.INC.B.W.L      ][IL.INC.B.W.L     ],
10197            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD       ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD       ]
10198
10199  .ICODE
10200       0B6: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;INCW
10201            IRD1[LOD][OS.MOD            ]                          [LOD][OS.MOD            ]
10202  .OCODE
10203       0B6: CNT0[NOP][IL.INC.B.W.L      ][IL.INC.B.W.L     ]  [NOP][IL.INC.B.W.L      ][IL.INC.B.W.L     ],
10204            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD       ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD       ]
10205
10206  .ICODE
10207       092: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;MCOMB
10208            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
10209  .OCODE
10210       092: CNT0[LOD][IL.MCOM.B.W.L.REG ][OS.WRT2          ]  [LOD][IL.MCOM.B.W.L.REG ][OS.WRT2          ],
10211            CNT1[NOP][IL.MCOM.B.W.L.MEM ][IL.MCOM.B.W.L.MEM]  [NOP][IL.MCOM.B.W.L.MEM ][IL.MCOM.B.W.L.MEM]
10212
10213  .ICODE
10214       0D2: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;MCOML
10215            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
10216  .OCODE
10217       0D2: CNT0[LOD][IL.MCOM.B.W.L.REG ][OS.WRT2          ]  [LOD][IL.MCOM.B.W.L.REG ][OS.WRT2          ],
10218            CNT1[NOP][IL.MCOM.B.W.L.MEM ][IL.MCOM.B.W.L.MEM]  [NOP][IL.MCOM.B.W.L.MEM ][IL.MCOM.B.W.L.MEM]
10219
10220  .ICODE
10221       0B2: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;MCOMW
10222            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
10223  .OCODE
10224       0B2: CNT0[LOD][IL.MCOM.B.W.L.REG ][OS.WRT2          ]  [LOD][IL.MCOM.B.W.L.REG ][OS.WRT2          ],
10225            CNT1[NOP][IL.MCOM.B.W.L.MEM ][IL.MCOM.B.W.L.MEM]  [NOP][IL.MCOM.B.W.L.MEM ][IL.MCOM.B.W.L.MEM]
```

```
                                                                H  4
;10226  .ICODE  ;       OPS             REG                     MEM             OPS         FPA REG                 FPA MEM
;10227       08E: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                      ;MNEGB
;10228            IRD1[LOD][OS.RED                ]                          [LOD][OS.RED                ]
;10229  .OCODE
;10230       08E: CNT0[LOD][IL.MNEG.B.W.L.REG     ][OS.WRT2               ]  [LOD][IL.MNEG.B.W.L.REG     ][OS.WRT2               ],
;10231            CNT1[NOP][IL.MNEG.B.W.L.MEM     ][IL.MNEG.B.W.L.MEM     ]  [NOP][IL.MNEG.B.W.L.MEM     ][IL.MNEG.B.W.L.MEM     ]
;10232
;10233  .ICODE
;10234       0CE: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                      ;MNEGL
;10235            IRD1[LOD][OS.RED                ]                          [LOD][OS.RED                ]
;10236  .OCODE
;10237       0CE: CNT0[LOD][IL.MNEG.B.W.L.REG     ][OS.WRT2               ]  [LOD][IL.MNEG.B.W.L.REG     ][OS.WRT2               ],
;10238            CNT1[NOP][IL.MNEG.B.W.L.MEM     ][IL.MNEG.B.W.L.MEM     ]  [NOP][IL.MNEG.B.W.L.MEM     ][IL.MNEG.B.W.L.MEM     ]
;10239
;10240  .ICODE
;10241       0AE: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                      ;MNEGW
;10242            IRD1[LOD][OS.RED                ]                          [LOD][OS.RED                ]
;10243  .OCODE
;10244       0AE: CNT0[LOD][IL.MNEG.B.W.L.REG     ][OS.WRT2               ]  [LOD][IL.MNEG.B.W.L.REG     ][OS.WRT2               ],
;10245            CNT1[NOP][IL.MNEG.B.W.L.MEM     ][IL.MNEG.B.W.L.MEM     ]  [NOP][IL.MNEG.B.W.L.MEM     ][IL.MNEG.B.W.L.MEM     ]
;10246
;10247  .ICODE
;10248       09E: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                      ;MOVAB
;10249            IRD1[LOD][OS.ADD                ]                          [LOD][OS.ADD                ]
;10250  .OCODE
;10251       09E: CNT0[LOD][IL.MOVA.B.W.L.REG     ][OS.WRT2               ]  [LOD][IL.MOVA.B.W.L.REG     ][OS.WRT2               ],
;10252            CNT1[NOP][IL.MOVA.B.W.L.MEM     ][IL.MOVA.B.W.L.MEM     ]  [NOP][IL.MOVA.B.W.L.MEM     ][IL.MOVA.B.W.L.MEM     ]
;10253
;10254  .ICODE
;10255       0DE: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                      ;MOVAL
;10256            IRD1[LOD][OS.ADD                ]                          [LOD][OS.ADD                ]
;10257  .OCODE
;10258       0DE: CNT0[LOD][IL.MOVA.B.W.L.REG     ][OS.WRT2               ]  [LOD][IL.MOVA.B.W.L.REG     ][OS.WRT2               ],
;10259            CNT1[NOP][IL.MOVA.B.W.L.MEM     ][IL.MOVA.B.W.L.MEM     ]  [NOP][IL.MOVA.B.W.L.MEM     ][IL.MOVA.B.W.L.MEM     ]
;10260
;10261  .ICODE
;10262       07E: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                      ;MOVAQ
;10263            IRD1[LOD][OS.ADD                ]                          [LOD][OS.ADD                ]
;10264  .OCODE
;10265       07E: CNT0[LOD][IL.MOVA.Q.REG         ][OS.WRT2               ]  [LOD][IL.MOVA.Q.REG         ][OS.WRT2               ],
;10266            CNT1[NOP][IL.MOVA.Q.MEM         ][IL.MOVA.Q.MEM         ]  [NOP][IL.MOVA.Q.MEM         ][IL.MOVA.Q.MEM         ]
;10267
;10268  .ICODE
;10269       03E: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                      ;MOVAW
;10270            IRD1[LOD][OS.ADD                ]                          [LOD][OS.ADD                ]
;10271  .OCODE
;10272       03E: CNT0[LOD][IL.MOVA.B.W.L.REG     ][OS.WRT2               ]  [LOD][IL.MOVA.B.W.L.REG     ][OS.WRT2               ],
;10273            CNT1[NOP][IL.MOVA.B.W.L.MEM     ][IL.MOVA.B.W.L.MEM     ]  [NOP][IL.MOVA.B.W.L.MEM     ][IL.MOVA.B.W.L.MEM     ]
;10274
;10275  .ICODE
;10276       090: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                      ;MOVB
;10277            IRD1[LOD][OS.RED                ]                          [LOD][OS.RED                ]
;10278  .OCODE
;10279       090: CNT0[LOD][IL.MOV.B.W.L.REG      ][OS.WRT2               ]  [LOD][IL.MOV.B.W.L.REG      ][OS.WRT2               ],
;10280            CNT1[NOP][IL.MOV.B.W.L.MEM      ][IL.MOV.B.W.L.MEM      ]  [NOP][IL.MOV.B.W.L.MEM      ][IL.MOV.B.W.L.MEM      ]
```

```
;10281  .ICODE  ;       OPS             REG                     MEM                     OPS         FPA REG                 FPA MEM
;10282       0D0: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC         ],                      ;MOVL
;10283            IRD1[LOD][OS.RED               ]                                   [LOD][OS.RED               ]
;10284  .OCODE
;10285       0D0: CNT0[LOD][IL.MOV.B.W.L.REG     ][OS.WRT2               ]  [LOD][IL.MOV.B.W.L.REG     ][OS.WRT2               ],
;10286            CNT1[NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]  [NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]
;10287
;10288  .ICODE
;10289       07D: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC         ],                      ;MOVQ
;10290            IRD1[LOD][OS.QRED              ]                                   [LOD][OS.QRED              ]
;10291  .OCODE
;10292       07D: CNT0[LOD][IL.MOVQ              ][OS.WRT2               ]  [LOD][IL.MOVQ              ][OS.WRT2               ],
;10293            CNT1[NOP][IL.MOVQ              ][IL.MOVQ               ]  [NOP][IL.MOVQ              ][IL.MOVQ               ]
;10294
;10295  .ICODE
;10296       0B0: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC         ],                      ;MOVW
;10297            IRD1[LOD][OS.RED               ]                                   [LOD][OS.RED               ]
;10298  .OCODE
;10299       0B0: CNT0[LOD][IL.MOV.B.W.L.REG     ][OS.WRT2               ]  [LOD][IL.MOV.B.W.L.REG     ][OS.WRT2               ],
;10300            CNT1[NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]  [NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]
;10301
;10302  .ICODE
;10303       09A: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC         ],                      ;MOVZBL
;10304            IRD1[LOD][OS.RED               ]                                   [LOD][OS.RED               ]
;10305  .OCODE
;10306       09A: CNT0[NOP][IL.MOVZ.BW.BL.WL     ][IL.MOVZ.BW.BL.WL      ]  [NOP][IL.MOVZ.BW.BL.WL     ][IL.MOVZ.BW.BL.WL      ],
;10307            CNT1[NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]  [NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]
;10308
;10309  .ICODE
;10310       09B: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC         ],                      ;MOVZBW
;10311            IRD1[LOD][OS.RED               ]                                   [LOD][OS.RED               ]
;10312  .OCODE
;10313       09B: CNT0[NOP][IL.MOVZ.BW.BL.WL     ][IL.MOVZ.BW.BL.WL      ]  [NOP][IL.MOVZ.BW.BL.WL     ][IL.MOVZ.BW.BL.WL      ],
;10314            CNT1[NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]  [NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]
;10315
;10316  .ICODE
;10317       03C: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC         ],                      ;MOVZWL
;10318            IRD1[LOD][OS.RED               ]                                   [LOD][OS.RED               ]
;10319  .OCODE
;10320       03C: CNT0[NOP][IL.MOVZ.BW.BL.WL     ][IL.MOVZ.BW.BL.WL      ]  [NOP][IL.MOVZ.BW.BL.WL     ][IL.MOVZ.BW.BL.WL      ],
;10321            CNT1[NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]  [NOP][IL.MOV.B.W.L.MEM     ][IL.MOV.B.W.L.MEM      ]
;10322
;10323  .ICODE
;10324       084: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC         ],                      ;MULB2
;10325            IRD1[LOD][OS.RED               ]                                   [LOD][OS.RED               ]
;10326  .OCODE
;10327       084: CNT0[LOD][OS.MOD               ][OS.MOD                ]  [LOD][OS.MOD               ][OS.MOD                ],
;10328            CNT1[NOP][IL.MUL2.B.W.L        ][IL.MUL2.B.W.L         ]  [NOP][FI.MUL2.B.W.L.MEM    ][FI.MUL2.B.W.L.MEM     ]
;10329
;10330  .ICODE
;10331       085: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC         ],                      ;MULB3
;10332            IRD1[LOD][OS.RED               ]                                   [LOD][OS.RED               ]
;10333  .OCODE
;10334       085: CNT0[LOD][OS.RED               ][OS.RED                ]  [LOD][OS.RED               ][OS.RED                ],
;10335            CNT1[NOP][IL.MUL3.B.W.L        ][IL.MUL3.B.W.L         ]  [NOP][FI.MUL3.B.W.L.MEM    ][FI.MUL3.B.W.L.MEM     ]
```

```
;10336  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;10337       0C4: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ],                    ;MULL2
;10338            IRD1[LOD][OS.RED            ]                        [LOD][OS.RED            ]
;10339  .OCODE
;10340       0C4: CNT0[LOD][OS.MOD            ][OS.MOD            ]  [LOD][OS.MOD            ][OS.MOD            ],
;10341            CNT1[NOP][IL.MUL2.B.W.L     ][IL.MUL2.B.W.L     ]  [NOP][FI.MUL2.B.W.L.MEM ][FI.MUL2.B.W.L.MEM ]
;10342
;10343  .ICODE
;10344       0C5: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ],                    ;MULL3
;10345            IRD1[LOD][OS.RED            ]                        [LOD][OS.RED            ]
;10346  .OCODE
;10347       0C5: CNT0[LOD][OS.RED            ][OS.RED            ]  [LOD][OS.RED            ][OS.RED            ],
;10348            CNT1[NOP][IL.MUL3.B.W.L     ][IL.MUL3.B.W.L     ]  [NOP][FI.MUL3.B.W.L.MEM ][FI.MUL3.B.W.L.MEM ]
;10349
;10350  .ICODE
;10351       0A4: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ],                    ;MULW2
;10352            IRD1[LOD][OS.RED            ]                        [LOD][OS.RED            ]
;10353  .OCODE
;10354       0A4: CNT0[LOD][OS.MOD            ][OS.MOD            ]  [LOD][OS.MOD            ][OS.MOD            ],
;10355            CNT1[NOP][IL.MUL2.B.W.L     ][IL.MUL2.B.W.L     ]  [NOP][FI.MUL2.B.W.L.MEM ][FI.MUL2.B.W.L.MEM ]
;10356
;10357  .ICODE
;10358       0A5: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ],                    ;MULW3
;10359            IRD1[LOD][OS.RED            ]                        [LOD][OS.RED            ]
;10360  .OCODE
;10361       0A5: CNT0[LOD][OS.RED            ][OS.RED            ]  [LOD][OS.RED            ][OS.RED            ],
;10362            CNT1[NOP][IL.MUL3.B.W.L     ][IL.MUL3.B.W.L     ]  [NOP][FI.MUL3.B.W.L.MEM ][FI.MUL3.B.W.L.MEM ]
;10363
;10364  .ICODE
;10365       09F: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ],                    ;PUSHAB
;10366            IRD1[LOD][OS.ADD            ]                        [LOD][OS.ADD            ]
;10367  .OCODE
;10368       09F: CNT0[NOP][IL.PUSHA.B.W.L    ][IL.PUSHA.B.W.L    ]  [NOP][IL.PUSHA.B.W.L    ][IL.PUSHA.B.W.L    ],
;10369            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;10370
;10371  .ICODE
;10372       0DF: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ],                    ;PUSHAL
;10373            IRD1[LOD][OS.ADD            ]                        [LOD][OS.ADD            ]
;10374  .OCODE
;10375       0DF: CNT0[NOP][IL.PUSHA.B.W.L    ][IL.PUSHA.B.W.L    ]  [NOP][IL.PUSHA.B.W.L    ][IL.PUSHA.B.W.L    ]  ,
;10376            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;10377
;10378  .ICODE
;10379       07F: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ],                    ;PUSHAQ
;10380            IRD1[LOD][OS.ADD            ]                        [LOD][OS.ADD            ]
;10381  .OCODE
;10382       07F: CNT0[NOP][IL.PUSHA.Q        ][IL.PUSHA.Q        ]  [NOP][IL.PUSHA.Q        ][IL.PUSHA.Q        ],
;10383            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;10384
;10385  .ICODE
;10386       03F: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ]                     ;PUSHAW
;10387            IRD1[LOD][OS.ADD            ]                        [LOD][OS.ADD            ]
;10388  .OCODE
;10389       03F: CNT0[NOP][IL.PUSHA.B.W.L    ][IL.PUSHA.B.W.L    ]  [NOP][IL.PUSHA.B.W.L    ][IL.PUSHA.B.W.L    ],
;10390            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
```

```
.ICODE   ;       OPS          REG                         MEM                   OP.        FPA REG                       FPA MEM
      0DD: FPD [NOP][IE.OPCOD.DEC          ]                                 [NOP][IE.OPCOD.DEC          ],                              ;PUSHL
           IRD1[LOD][OS.RED                ]                                 [LOD][OS.RED                ]
.OCODE
      0DD: CNT0[NOP][IL.PUSHL              ][IL.PUSHL                ]       [NOP][IL.PUSHL              ][IL.PUSHL                ],
           CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD              ]       [NOP][IE.BAD.IRD            ][IE.BAD.IRD              ]

.ICODE
      09C: FPD [NOP][IE.OPCOD.DEC          ]                                 [NOP][IE.OPCOD.DEC          ],                              ;ROTL
           IRD1[LOD][OS.RED                ]                                 [LOD][OS.RED                ]
.OCODE
      09C: CNT0[LOD][OS.RED                ][OS.RED                  ]       [LOD][OS.RED                ][OS.RED                  ],
           CNT1[LOD][IL.ROTLREG            ][IL.ROTLMEM              ]       [LOD][IL.ROTLREG            ][IL.ROTLMEM              ]

.ICODE
      0D9: FPD [NOP][IE.OPCOD.DEC          ]                                 [NOP][IE.OPCOD.DEC          ],                              ;SBWC
           IRD1[LOD][OS.RED                ]                                 [LOD][OS.RED                ]
.OCODE
      0D9: CNT0[LOD][IL.SBWCREG            ][OS.MOD                  ]       [LOD][IL.SBWCREG            ][OS.MOD                  ],
           CNT1[NOP][IL.SBWCMEM            ][IL.SBWCMEM              ]       [NOP][IL.SBWCMEM            ][IL.SBWCMEM              ]

.ICODE
      082: FPD [NOP][IE.OPCOD.DEC          ]                                 [NOP][IE.OPCOD.DEC          ],                              ;SUBB2
           IRD1[LOD][OS.RED                ]                                 [LOD][OS.RED                ]
.OCODE
      082: CNT0[LOD][IL.SUB2.B.W.L.REG     ][OS.MOD                  ]       [LOD][IL.SUB2.B.W.L.REG     ][OS.MOD                  ],
           CNT1[NOP][IL.SUB2.B.W.L.MEM     ][IL.SUB2.B.W.L.MEM       ]       [NOP][IL.SUB2.B.W.L.MEM     ][IL.SUB2.B.W.L.MEM       ]

.ICODE
      083: FPD [NOP][IE.OPCOD.DEC          ]                                 [NOP][IE.OPCOD.DEC          ],                              ;SUBB3
           IRD1[LOD][OS.RED                ]                                 [LOD][OS.RED                ]
.OCODE
      083: CNT0[LOD][OS.RED                ][OS.RED                  ]       [LOD][OS.RED                ][OS.RED                  ],
           CNT1[LOD][IL.SUB3.B.W.L.REG     ][IL.SUB3.B.W.L.MEM       ]       [LOD][IL.SUB3.B.W.L.REG     ][IL.SUB3.B.W.L.MEM       ]

.ICODE
      0C2: FPD [NOP][IE.OPCOD.DEC          ]                                 [NOP][IE.OPCOD.DEC          ],                              ;SUBL2
           IRD1[LOD][OS.RED                ]                                 [LOD][OS.RED                ]
.OCODE
      0C2: CNT0[LOD][IL.SUB2.B.W.L.REG     ][OS.MOD                  ]       [LOD][IL.SUB2.B.W.L.REG     ][OS.MOD                  ],
           CNT1[NOP][IL.SUB2.B.W.L.MEM     ][IL.SUB2.B.W.L.MEM       ]       [NOP][IL.SUB2.B.W.L.MEM     ][IL.SUB2.B.W.L.MEM       ]

.ICODE
      0C3: FPD [NOP][IE.OPCOD.DEC          ]                                 [NOP][IE.OPCOD.DEC          ],                              ;SUBL3
           IRD1[LOD][OS.RED                ]                                 [LOD][OS.RED                ]
.OCODE
      0C3: CNT0[LOD][OS.RED                ][OS.RED                  ]       [LOD][OS.RED                ][OS.RED                  ],
           CNT1[LOD][IL.SUB3.B.W.L.REG     ][IL.SUB3.B.W.L.MEM       ]       [LOD][IL.SUB3.B.W.L.REG     ][IL.SUB3.B.W.L.MEM       ]

.ICODE
      0A2: FPD [NOP][IE.OPCOD.DEC          ]                                 [NOP][IE.OPCOD.DEC          ],                              ;SUBW2
           IRD1[LOD][OS.RED                ]                                 [LOD][OS.RED                ]
.OCODE
      0A2: CNT0[LOD][IL.SUB2.B.W.L.REG     ][OS.MOD                  ]       [LOD][IL.SUB2.B.W.L.REG     ][OS.MOD                  ],
           CNT1[NOP][IL.SUB2.B.W.L.MEM     ][IL.SUB2.B.W.L.MEM       ]       [NOP][IL.SUB2.B.W.L.MEM     ][IL.SUB2.B.W.L.MEM       ]
```

```
;10446  .ICODE  ;      OPS          REG                    MEM                    OPS          FPA REG                FPA MEM
;10447       0A3: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                          ;SUBW3
;10448            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;10449  .OCODE
;10450       0A3: CNT0[LOD][OS.RED            ][OS.RED                ] [LOD][OS.RED            ][OS.RED                ],
;10451            CNT1[LOD][IL.SUB3.B.W.L.REG ][IL.SUB3.B.W.L.MEM     ] [LOD][IL.SUB3.B.W.L.REG ][IL.SUB3.B.W.L.MEM     ]
;10452
;10453  .ICODE
;10454       095: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                          ;TSTB
;10455            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;10456  .OCODE
;10457       095: CNT0[NOP][IL.TST.B.W.L      ][IL.TST.B.W.L          ] [NOP][IL.TST.B.W.L      ][IL.TST.B.W.L          ],
;10458            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD            ] [NOP][IE.BAD.IRD        ][IE.BAD.IRD            ]
;10459
;10460  .ICODE
;10461       0D5: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                          ;TSTL
;10462            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;10463  .OCODE
;10464       0D5: CNT0[NOP][IL.TST.B.W.L      ][IL.TST.B.W.L          ] [NOP][IL.TST.B.W.L      ][IL.TST.B.W.L          ],
;10465            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD            ] [NOP][IE.BAD.IRD        ][IE.BAD.IRD            ]
;10466
;10467  .ICODE
;10468       0B5: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                          ;TSTW
;10469            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;10470  .OCODE
;10471       0B5: CNT0[NOP][IL.TST.B.W.L      ][IL.TST.B.W.L          ] [NOP][IL.TST.B.W.L      ][IL.TST.B.W.L          ],
;10472            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD            ] [NOP][IE.BAD.IRD        ][IE.BAD.IRD            ]
;10473
;10474  .ICODE
;10475       08C: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                          ;XORB2
;10476            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;10477  .OCODE
;10478       08C: CNT0[LOD][IL.XOR2.B.W.L.REG ][OS.MOD                ] [LOD][IL.XOR2.B.W.L.REG ][OS.MOD                ],
;10479            CNT1[NOP][IL.XOR2.B.W.L.MEM ][IL.XOR2.B.W.L.MEM     ] [NOP][IL.XOR2.B.W.L.MEM ][IL.XOR2.B.W.L.MEM     ]
;10480
;10481  .ICODE
;10482       08D: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                          ;XORB3
;10483            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;10484  .OCODE
;10485       08D: CNT0[LOD][OS.RED            ][OS.RED                ] [LOD][OS.RED            ][OS.RED                ],
;10486            CNT1[LOD][IL.XOR3.B.W.L.REG ][IL.XOR3.B.W.L.MEM     ] [LOD][IL.XOR3.B.W.L.REG ][IL.XOR3.B.W.L.MEM     ]
;10487
;10488  .ICODE
;10489       0CC: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                          ;XORL2
;10490            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;10491  .OCODE
;10492       0CC: CNT0[LOD][IL.XOR2.B.W.L.REG ][OS.MOD                ] [LOD][IL.XOR2.B.W.L.REG ][OS.MOD                ],
;10493            CNT1[NOP][IL.XOR2.B.W.L.MEM ][IL.XOR2.B.W.L.MEM     ] [NOP][IL.XOR2.B.W.L.MEM ][IL.XOR2.B.W.L.MEM     ]
;10494
;10495  .ICODE
;10496       0CD: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                          ;XORL3
;10497            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;10498  .OCODE
;10499       0CD: CNT0[LOD][OS.RED            ][OS.RED                ] [LOD][OS.RED            ][OS.RED                ],
;10500            CNT1[LOD][IL.XOR3.B.W.L.REG ][IL.XOR3.B.W.L.MEM     ] [LOD][IL.XOR3.B.W.L.REG ][IL.XOR3.B.W.L.MEM     ]
```

```
;10501  .ICODE  ;      OPS          REG                          MEM                OPS        FPA REG                FPA MEM
;10502        0AC: FPD [NOP][IE.OPCOD.DEC        ]                              [NOP][IE.OPCOD.DEC        ],                    ;XORW2
;10503             IRD1[LOD][OS.RED              ]                              [LOD][OS.RED              ]
;10504  .OCODE
;10505        0AC: CNT0[LOD][IL.XOR2.B.W.L.REG   ][OS.MOD                ]  [LOD][IL.XOR2.B.W.L.REG   ][OS.MOD                ],
;10506             CNT1[NOP][IL.XOR2.B.W.L.MEM   ][IL.XOR2.B.W.L.MEM   ]  [NOP][IL.XOR2.B.W.L.MEM   ][IL.XOR2.B.W.L.MEM   ]
;10507
;10508  .ICODE
;10509        0AD: FPD [NOP][IE.OPCOD.DEC        ]                              [NOP][IE.OPCOD.DEC        ],                    ;XORW3
;10510             IRD1[LOD][OS.RED              ]                              [LOD][OS.RED              ]
;10511  .OCODE
;10512        0AD: CNT0[LOD][OS.RED              ][OS.RED                ]  [LOD][OS.RED              ][OS.RED                ],
;10513             CNT1[LOD][IL.XOR3.B.W.L.REG   ][IL.XOR3.B.W.L.MEM   ]  [LOD][IL.XOR3.B.W.L.REG   ][IL.XOR3.B.W.L.MEM   ]
```

```
;10514  .TOC   "      Integer, Logical, & Address    :  Dsize Rom Definition"
;10515  .DCODE
;10516
;10517  058:    SI?E [WORD] [WORD] [   0] [   0] [   0] [   0]  ;ADAWI
;10518  080:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;ADDB2
;10519  081:    SIZE [BYTE] [BYTE] [BYTE] [   0] [   0] [   0]  ;ADDB3
;10520  0C0:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;ADDL2
;10521  0C1:    SIZE [LONG] [LONG] [LONG] [   0] [   0] [   0]  ;ADDL3
;10522  0A0:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;ADDW2
;10523  0A1:    SIZE [WORD] [WORD] [WORD] [   0] [   0] [   0]  ;ADDW3
;10524  0D8:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;ADWC
;10525  078:    SIZE [BYTE] [LONG] [LONG] [   0] [   0] [   0]  ;ASHL
;10526  079:    SIZE [BYTE] [QUAD] [QUAD] [   0] [   0] [   0]  ;ASHQ
;10527  08A:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;BICB2
;10528  08B:    SIZE [BYTE] [BYTE] [BYTE] [   0] [   0] [   0]  ;BICB3
;10529  0CA:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;BICL2
;10530  0CB:    SIZE [LONG] [LONG] [LONG] [   0] [   0] [   0]  ;BICL3
;10531  0AA:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;BICW2
;10532  0AB:    SIZE [WORD] [WORD] [WORD] [   0] [   0] [   0]  ;BICW3
;10533  088:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;BISB2
;10534  089:    SIZE [BYTE] [BYTE] [BYTE] [   0] [   0] [   0]  ;BISB3
;10535  0C8:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;BISL2
;10536  0C9:    SIZE [LONG] [LONG] [LONG] [   0] [   0] [   0]  ;BISL3
;10537  0A8:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;BISW2
;10538  0A9:    SIZE [WORD] [WORD] [WORD] [   0] [   0] [   0]  ;BISW3
;10539  093:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;BITB
;10540  0D3:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;BITL
;10541  0B3:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;BITW
;10542  094:    SIZE [BYTE] [   0] [   0] [   0] [   0] [   0]  ;CLRB
;10543  0D4:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;CLRL
;10544  07C:    SIZE [QUAD] [   0] [   0] [   0] [   0] [   0]  ;CLRQ
;10545  0B4:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;CLRW
;10546  091:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;CMPB
;10547  0D1:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;CMPL
;10548  0B1:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;CMPW
;10549  098:    SIZE [BYTE] [LONG] [   0] [   0] [   0] [   0]  ;CVTBL
;10550  099:    SIZE [BYTE] [WORD] [   0] [   0] [   0] [   0]  ;CVTBW
;10551  0F6:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;CVTLB
;10552  0F7:    SIZE [LONG] [WORD] [   0] [   0] [   0] [   0]  ;CVTLW
;10553  033:    SIZE [WORD] [BYTE] [   0] [   0] [   0] [   0]  ;CVTWB
;10554  032:    SIZE [WORD] [LONG] [   0] [   0] [   0] [   0]  ;CVTWL
;10555  097:    SIZE [BYTE] [   0] [   0] [   0] [   0] [   0]  ;DECB
;10556  0D7:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;DECL
;10557  0B7:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;DECW
;10558  086:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;DIVB2
;10559  087:    SIZE [BYTE] [BYTE] [BYTE] [   0] [   0] [   0]  ;DIVB3
;10560  0C6:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;DIVL2
;10561  0C7:    SIZE [LONG] [LONG] [LONG] [   0] [   0] [   0]  ;DIVL3
;10562  0A6:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;DIVW2
;10563  0A7:    SIZE [WORD] [WORD] [WORD] [   0] [   0] [   0]  ;DIVW3
;10564  07B:    SIZE [LONG] [QUAD] [LONG] [LONG] [   0] [   0]  ;EDIV
;10565  07A:    SIZE [LONG] [LONG] [LONG] [QUAD] [   0] [   0]  ;EMUL
;10566  096:    SIZE [BYTE] [   0] [   0] [   0] [   0] [   0]  ;INCB
;10567  0D6:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;INCL
;10568  0B6:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;INCW
```

```
;10569  092:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;MCOMB
;10570  0D2:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;MCOML
;10571  0B2:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;MCOMW
;10572  08E:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;MNEGB
;10573  0CE:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;MNEGL
;10574  0AE:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;MNEGW
;10575  09E:    SIZE [BYTE] [LONG] [   0] [   0] [   0] [   0]  ;MOVAB
;10576  0DE:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;MOVAL
;10577  07E:    SIZE [QUAD] [LONG] [   0] [   0] [   0] [   0]  ;MOVAQ
;10578  03E:    SIZE [WORD] [LONG] [   0] [   0] [   0] [   0]  ;MOVAW
;10579  090:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;MOVB
;10580  0D0:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;MOVL
;10581  07D:    SIZE [QUAD] [QUAD] [   0] [   0] [   0] [   0]  ;MOVQ
;10582  0B0:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;MOVW
;10583  09A:    SIZE [BYTE] [LONG] [   0] [   0] [   0] [   0]  ;MOVZBL
;10584  09B:    SIZE [BYTE] [WORD] [   0] [   0] [   0] [   0]  ;MOVZBW
;10585  03C:    SIZE [WORD] [LONG] [   0] [   0] [   0] [   0]  ;MOVZWL
;10586  084:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;MULB2
;10587  085:    SIZE [BYTE] [BYTE] [BYTE] [   0] [   0] [   0]  ;MULB3
;10588  0C4:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;MULL2
;10589  0C5:    SIZE [LONG] [LONG] [LONG] [   0] [   0] [   0]  ;MULL3
;10590  0A4:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;MULW2
;10591  0A5:    SIZE [WORD] [WORD] [WORD] [   0] [   0] [   0]  ;MULW3
;10592  09F:    SIZE [BYTE] [   0] [   0] [   0] [   0] [   0]  ;PUSHAB
;10593  0DF:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;PUSHAL
;10594  07F:    SIZE [QUAD] [   0] [   0] [   0] [   0] [   0]  ;PUSHAQ
;10595  03F:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;PUSHAW
;10596  0DD:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;PUSHL
;10597  09C:    SIZE [BYTE] [LONG] [LONG] [   0] [   0] [   0]  ;ROTL
;10598  0D9:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;SBWC
;10599  082:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;SUBB2
;10600  083:    SIZE [BYTE] [BYTE] [BYTE] [   0] [   0] [   0]  ;SUBB3
;10601  0C2:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;SUBL2
;10602  0C3:    SIZE [LONG] [LONG] [LONG] [   0] [   0] [   0]  ;SUBL3
;10603  0A2:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;SUBW2
;10604  0A3:    SIZE [WORD] [WORD] [WORD] [   0] [   0] [   0]  ;SUBW3
;10605  095:    SIZE [BYTE] [   0] [   0] [   0] [   0] [   0]  ;TSTB
;10606  0D5:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;TSTL
;10607  0B5:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;TSTW
;10608  08C:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;XORB2
;10609  08D:    SIZE [BYTE] [BYTE] [BYTE] [   0] [   0] [   0]  ;XORB3
;10610  0CC:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;XORL2
;10611  0CD:    SIZE [LONG] [LONG] [LONG] [   0] [   0] [   0]  ;XORL3
;10612  0AC:    SIZE [WORD] [WORD] [   0] [   0] [   0] [   0]  ;XORW2
;10613  0AD:    SIZE [WORD] [WORD] [WORD] [   0] [   0] [   0]  ;XORW3
;10614
;10615  .UCODE
                                        ;10616 .BIN
```

10616; This page intentionally left blank.

```
                                        ;10617 .TOC "FLOAT.MIC"
                                        ;10618 .TOC "Revision 26.0"
                                        ;10619 ;       Gerard Koeckhoven, Audrey Reith
;10620  .NOBIN
;10621  .TOC "  Revision History"
;10622
;10623  ; 26    Fix FPA interface for unpacking short literal
;10624  ; 25    POLYF and POLYD have problem if 1st coefficient or any part product is 0
;10625  ;       because the result of the multiply step did not clear the
;10626  ;       working exp field
;10627  ; 24    Fix a POLYD problem negative part product underflowed and next
;10628  ;       coefficient is 0
;10629  ; 23    Fix a POLYD problem that causes the VA to be incorrect, when the arg =0,
;10630  ;       and the degree not equal to zero and doing a read with FPD set.
;10631  ;       Recover locations 232 and 233 for use in IANDE.MIC
;10632  ;       Assign permanent addresses to free locations (where possible)
;10633  ; 22    Use locations 232 and 233 for a fix in IANDE but leave them in FLOAT.
;10634  ;       Fix a ADDD,SUBD,EMODD problem, when doing any of these instructions
;10635  ;       and the high order number is zero before the normalization then the
;10636  ;       9 high order bits of the low order longword will be missing.
;10637  ;       Fix a POLYD and POLYF problem , which causes the fraction part
;10638  ;       of the result to be unpredictable when:
;10639  ;               a - the product step with exp(biased) = 0  and
;10640  ;               b - the fraction is between 1/4 and 1/2 and
;10641  ;               c - the coefficient exponent (biased) > 31
;10642  ; 21    Fix EMODD creating dirty zero if extreme overflow.
;10643  ;       Add some labels for G and H.
;10644  ; 20    FIX POLYD ACCURACY AND DIRTY ZERO BUGS.
;10645  ;       CHANGE CRC TO CLEAR PSL<C>.
;10646  ;       CHANGE TO ACBF AND ACBD FOR G AND H FLOAT TO SHARE.
;10647  ;       Fix bad litrl in POLYF and POLYD FPA interface.
;10648  ;       Fix setting of CC in FI.POLYFD.
;10649  ;       Add branch on regmode to FI.EMODD.
;10650  ;       Fix FPA interface for POLYF and POLYD.
;10651  ;       Fix FI.CMPD condition codes.
;10652  ;       Fix size on FPA converts.
;10653  ; 19    Initial release.
                                        ;10654          .BIN
```

```
                                ;10655 .TOC "  Floating point and CRC          : MOVF"
                                ;10656
                                ;10657 ;*************************************************************************
                                ;10658 ;       50      MOVF    src.rf,dst.wf
                                ;10659 ;
                                ;10660 ;
                                ;10661 ;               Input   (dst.wf in memory)      Q               src.rf
                                ;10662 ;                                               VA              dst.wf
                                ;10663 ;
                                ;10664 ;               Input   (dst.wf in register)    MDR             src.rf
                                ;10665 ;                                               RNUM            dst.wf
                                ;10666 ;
                                ;10667 ;               Operation:
                                ;10668 ;
                                ;10669 ;                       Validate floating point operand (not reserved)
                                ;10670 ;                       Write to the dst.wf and set condition codes
                                ;10671 ;
                                ;10672 ;       **********NOT AN FPA INSTRUCTION**********
                                ;10673 ;
                                ;10674 ;*************************************************************************
                                ;10675
                                ;10676 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;10677 FP.MOVF.MEM:                            ; ********************
                                ;10678         ;-------------------------------;
U 030D, 0480,003A,403D,8467,0031,1 ;10679      MDR_Q,NEXT/FP.MOVF.REG          ;  GET FP VALUE TO MBUS SOURCE
                                ;10680
                                ;10681 FP.MOVF.REG:                            ; ********************
                                ;10682         ;-------------------------------;
                                ;10683         WB_EXP(M[MDR]),                 ; EXPONENT TEST FOR
U 0311, 0481,2437,0AB0,0047,084F,C 323* ;10684 FRO.FLTZ?                       ; >,<,=0,RESERVED
                                ;10685
                                ;10686 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;10687 =00
                                ;10688         ;00-------------------------------; RESERVED OPERAND
U 04FC, 0C80,0036,4030,0047,00FF,8 ;10689      NEXT/IE.OPER.FAULT              ;
                                ;10690
                                ;10691         ;01-------------------------------; SRC < 0
                                ;10692         R[DST.R]_M[MDR],                ; WRITE TO MEMORY OR REGISTER
                                ;10693         WRITE NOTREG,SIZE[LONG],        ; SET CONDITION CODES
U 04FD, 0C85,2592,412C,55DA,003F,9 ;10694      CCOP2,IRD1                      ; END OF INSTRUCTION
                                ;10695
                                ;10696         ;10-------------------------------; SRC=0
                                ;10697         R[DST.R]_0,                     ; BE SURE IS CLEAN ZERO
                                ;10698         WRITE NOTREG,SIZE[LONG],        ; WRITE IT
U 04FE, 0C84,05B7,012C,55DA,003F,9 ;10699      CCOP2,IRD1                      ; END OF INSTRUCTION
                                ;10700
                                ;10701         ;11-------------------------------; SRC > 0
                                ;10702         R[DST.R]_M[MDR],                ; WRITE VALUE
                                ;10703         WRITE NOTREG,SIZE[LONG],        ; SET CONDITION CODES
U 04FF, 0C85,2592,412C,55DA,003F,9 ;10704      CCOP2,IRD1                      ; END OF INSTRUCTION
```

```
                                ;10705 .TOC "  Floating point and CRC          : MOVD"
                                ;10706
                                ;10707 ;*************************************************************************
                                ;10708 ;
                                ;10709 ;       70      MOVD    src.rd,dst.wd
                                ;10710 ;
                                ;10711 ;
                                ;10712 ;               Input   (dst.wd in memory)      Temp1,Q         src.rd
                                ;10713 ;                                               VA              dst.wd
                                ;10714 ;
                                ;10715 ;               Input   (dst.wd in register)    Temp1,MDR       src.rd
                                ;10716 ;                                               RNUM            dst.wd
                                ;10717 ;
                                ;10718 ;               Operation:
                                ;10719 ;
                                ;10720 ;                       Validate floating point operand (not reserved)
                                ;10721 ;                       Write to dst.wd and set condition codes
                                ;10722 ;
                                ;10723 ;       **********NOT AN FPA INSTRUCTION**********
                                ;10724 ;
                                ;10725 ;
                                ;10726 ;*************************************************************************
                                ;10727
                                ;10728 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;10729 FP.MOVD.MEM:                             ; ********************
                                ;10730 FP.MNEGD.MEM:                            ; ********************
                                ;10731         ;-------------------------------;
                                ;10732         MDR_Q,                          ; MOVE BITS32-63 OF OPERAND BACK TO MDR
U 0315, 0080,003A,467D,8467,0006,0 ;10733      IR<2-0>?,NEXT/FP.MOVD.REG       ; WAS IT MOVD OR MNEGD
                                ;10734
                                ;10735 =000
                                ;10736 FP.MOVD.REG:                             ; ********************
                                ;10737         ;000-------------------------------;
                                ;10738         WB_EXP(M[TEMP1]),               ; TEST OPERAND FOR >,<,= 0,RESERVED
                                ;10739         SET WRITE(FLAG1),               ; SET FLAG 1 FOR LATER WRITE TEST
U 0060, 0048,1437,0ABC,0047,0850,0 323* ;10740 FRO.FLTZ?,NEXT/FP.MOVD.10        ;
                                ;10741
                                ;10742 =010
                                ;10743 FP.MNEGD.REG:                            ; ********************
                                ;10744         ;010-------------------------------;
                                ;10745         WB_EXP(M[TEMP1]),               ; TEST OPERAND FOR >,<,= 0,RESERVED
                                ;10746         SET WRITE(FLAG1),               ; SET FLAG1 FOR LATER WRITE TEST
                                ;10747         FRO.FLTZ?,PUSH,                 ;
U 0062, 0048,1437,0AB0,0047,0052,0 323* ;10748 NEXT/FP.MNEGD.10                 ;
                                ;10749
                                ;10750         ;011-------------------------------;
                                ;10751         R[DST.R]_M[TEMP1],              ; FIRST WORD WRITTEN OUT
                                ;10752         CCOP1.SET MM.NOINT,             ; CONDITION CODES NO INTERRUPTS
                                ;10753         WRITE NOTREG,SIZE[IDEP],        ; WRITE IT OUT
U 0063, 0064,1592,403C,4DDA,0056,F ;10754      NEXT/FP.WRITE.SEC               ; JOIN COMMON FLOWS
                                ;10755 ;
```

```
                                        ;10756 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                        ;10757 =00
                                        ;10758 FP.MOVD.10:
                                        ;10759          ;00-------------------------------; RESERVED OPERAND
U 0500, 0C80,0036,4030,0047,00FF,8      ;10760          NEXT/IE.OPER.FAULT                ;
                                        ;10761
                                        ;10762          ;01-------------------------------; SRC < 0
                                        ;10763          R[DST.R]_M[TEMP1                  ; WRITE FIRST PART OF SRC
                                        ;10764          CCOP2,                            ; SET CONDITION CODES
                                        ;10765          WRITE NOTREG,SIZE[IDEP],          ;
                                        ;10766          SET MM.NOINT,                     ; NO INTERRRUPTS BETWEEN WRITES
U 0501, 0064,1592,403C,55DA,0056,F      ;10767          NEXT/FP.WRITE.SEC                 ; CONTINUE WITH SECOND HALF OF SRC
                                        ;10768
                                        ;10769          ;10-------------------------------; SRC = 0
U 0502, 0884,05B7,0030,44E7,0050,3      ;10770          R[TEMP1]_0,MDR_0                  ; BE SURE CLEAN ZERO BOTH
                                        ;10771
                                        ;10772          ;11-------------------------------; SRC > 0
                                        ;10773          R[DST.R]_M[TEMP1],                ; WRITE FIRST PART OF SRC
                                        ;10774          CCOP2,                            ; SET CONDITION CODES
                                        ;10775          WRITE NOTREG,SIZE[IDEP],          ;
                                        ;10776          SET MM.NOINT,                     ; NO INTERRUPTS BETWEEN WRITES
U 0503, 0064,1592,403C,55DA,0056,F      ;10777          NEXT/FP.WRITE.SEC                 ; CONTINUE WRITE
```

```
                                ;10778 .TOC "  Floating point and CRC          : MNEGF"
                                ;10779
                                ;10780 ;*****************************************************************************
                                ;10781 ;
                                ;10782 ;
                                ;10783 ;       52      MNEGF   src.rf,dst.wf
                                ;10784 ;
                                ;10785 ;
                                ;10786 ;               Input   (dst.wf in memory)      Q       src.rf
                                ;10787 ;                                               VA      dst.wf
                                ;10788 ;
                                ;10789 ;               Input   (dst.wf in register)    MDR     src.rf
                                ;10790 ;                                               RNUM    dst.wf
                                ;10791 ;
                                ;10792 ;               Operation:
                                ;10793 ;
                                ;10794 ;                       Validate floating point operand (not reserved)
                                ;10795 ;                       Invert sign bit, Write to dst.wf
                                ;10796 ;                       Set condition codes
                                ;10797 ;
                                ;10798 ;       **********NOT AN FPA INSTRUCTION**********
                                ;10799 ;
                                ;10800 ;*****************************************************************************
                                ;10801 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;10802 FP.MNEGF.MEM:                           ; ********************
                                ;10803         ;-------------------------------;
U 031D, 0C80,003A,403D,8467,0032,5 ;10804      MDR_Q                           ; GET SRC TO AN MBUS
                                ;10805
                                ;10806 FP.MNEGF.REG:                           ; ********************
                                ;10807         ;-------------------------------;
U 0325, 0C81,2437,0AB0,0047,0850,C 323* ;10808 WB_EXP(M[MDR]),FRO.FLTZ?         ; TEST FOR VALIDITY OF OPERAND
                                ;10809
                                ;10810 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;10811 =00
                                ;10812         ;00-------------------------------; RESERVED OPERAND
U 050C, 0C80,0036,4030,0047,00FF,8 ;10813      NEXT/IE.OPER.FAULT
                                ;10814
                                ;10815         ;01-------------------------------; SRC < 0
                                ;10816         MDR_M[MDR].XOR.ZLIT12[8],       ; INVERT THE SIGN BIT
U 050D, 0581,2D53,4030,4467,004F,A ;10817      NEXT/FP.MNEGF.15                ; JOIN COMMON WRITE
                                ;10818
                                ;10819         ;10-------------------------------; SRC = 0
                                ;10820         R[DST.R]_0,                     ; BE SURE ITS CLEAN ZERO
                                ;10821         WRITE NOTREG,SIZE[LONG],        ; WRITE OUT CLEAN ZERO
U 050E, 0C84,05B7,012C,4DDA,003F,9 ;10822      CCOP1,IRD1                      ; END OF INSTRUCTION
                                ;10823
                                ;10824         ;11-------------------------------; SRC >0
U 050F, 0581,2D53,4030,4467,004F,A ;10825      MDR_M[MDR].XOR.ZLIT12[8]        ; INVERT THE SIGN BIT
                                ;10826
                                ;10827 FP.MNEGF.15:
                                ;10828         ;-------------------------------;
                                ;10829         R[DST.R]_M[MDR],                ; SET CONDITION CODES
                                ;10830         WRITE NOTREG,SIZE[LONG],        ;
U 04FA, 0C85,2592,412C,4DDA,003F,9 ;10831      CCOP1,IRD1                      ; END OF INSTRUCTION
```

```
                                        .TOC "  Floating point and CRC          : MNEGD"

                                        ;***********************************************************************
                                        ;
                                        ;
                                        ;       72      MNEGD   src.rd,dst.wd
                                        ;
                                        ;
                                        ;               *Input  (dst.wd in memory)      Temp1,Q         src.rd
                                        ;                                               VA              dst.wd
                                        ;
                                        ;               *Input  (dst.wd in register)    Temp1,MDR       src.rd
                                        ;                                               RNUM            dst.wd
                                        ;
                                        ;
                                        ;               Operation:
                                        ;
                                        ;                       Validate floating point operand (not reserved)
                                        ;                       Invert sign bit
                                        ;                       Write to dst.wd and set condition codes
                                        ;
                                        ;       **********NOT AN FPA INSTRUCTION**********
                                        ;
                                        ;                       * Entry on page with MOVD instruction
                                        ;
                                        ;
                                        ;***********************************************************************

                                        =00
                                        FX.MNEGGH.10:                                   ; USED FOR GH HOOK ALSO
                                        FP.MNEGD.10:
                                                ;00-----------------------------; RESERVED OPERAND
U 052C, 0C80,0036,4030,0047,00FF,8              NEXT/IE.OPER.FAULT              ;

                                                ;01-----------------------------; SRC < 0
                                                M[TEMP1]_MB.XOR.ZLIT12[8],      ; INVERT THE SIGN BIT
U 052D, 0586,1D53,40B0,4047,0000,1              RETURN [+1]                     ;

                                        FX.MNEGGH.20:                                   ; SPECIAL ENTRY G,H
                                                ;10-----------------------------; SRC = 0
                                                R[TEMP1]_0,                     ; BE SURE ITS TWO CLEAN WORDS OF ZERO
                                                MDR_0,                          ; CLEAN UP SECOND WORD
                                                SET WRITE(FLAG1),               ; FLAG ALREADY SET FOR MNEGD
U 052E, 0C4C,05B7,00B0,44E7,0000,1              RETURN [+1]                     ; BEING DONE FOR OTHER CODE

                                                ;11-----------------------------; SRC > 0
                                                M[TEMP1]_MB.XOR.ZLIT12[8],      ; INVERT THE SIGN BIT
U 052F, 0586,1D53,40B0,4047,0000,1              RETURN [+1]                     ;
```

```
                        ;10880 .TOC "  Floating point and CRC          : TSTF & TSTD"
                        ;10881
                        ;10882 ;*************************************************************************
                        ;10883 ;
                        ;10884 ;
                        ;10885 ;       53      TSTF    src.rf
                        ;10886 ;
                        ;10887 ;
                        ;10888 ;               Input                   MDR             src.rf
                        ;10889 ;
                        ;10890 ;
                        ;10891 ;       73      TSTD    src.rd
                        ;10892 ;
                        ;10893 ;
                        ;10894 ;               Input                   Temp1,MDR       src.rd
                        ;10895 ;
                        ;10896 ;
                        ;10897 ;               Operation:
                        ;10898 ;
                        ;10899 ;                       Validate floating point operand (not reserved)
                        ;10900 ;                       Set condition code
                        ;10901 ;
                        ;10902 ;       **********NOT  FPA INSTRUCTIONS**********
                        ;10903 ;
                        ;10904 ;*************************************************************************
                        ;10905
                        ;10906 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;10907 FP.TSTF:                                 ; ********************
                        ;10908         ;-------------------------------;
                        ;10909         WB_EXP(M[MDR]),                 ; TEST OPERAND FOR VALIDITY
U 0329, 0481,2437,0AB0,0047,0856,4 323* ;10910         FRD.FLTZ?,NEXT/FP.TSTF.10       ; LOOK FOR RESERVED OPERAND
                        ;10911
                        ;10912 FP.TSTD:                                 ; ********************
                        ;10913         ;-------------------------------;
                        ;10914         WB_EXP(M[TEMP1]),               ; TEST OPERAND FOR VALIDITY
U 032D, 0080,1437,0AB0,0047,0856,4 323* ;10915         FRD.FLTZ?,NEXT/FP.TSTF.10       ; LOOK FOR RESERVED OPERAND
                        ;10916
                        ;10917 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                        ;10918 =00
                        ;10919 FX.TSTGH.10:                             ; HOOK FOR G AND H
                        ;10920 FP.TSTF.10:
                        ;10921         ;00-------------------------------; RESERVED OPERAND
U 0564, 0C80,0036,4030,0047,00FF,8      ;10922         NEXT/IE.OPER.FAULT              ;
                        ;10923
                        ;10924         ;01-------------------------------; SRC < 0
                        ;10925         CC_ZLITO[8],                    ; SET CONDITION CODES
U 0565, 0580,0C37,0130,40A7,003F,9      ;10926         IRD1                            ; END OF INSTRUCTION
                        ;10927
                        ;10928         ;10-------------------------------; SRC = 0
                        ;10929         CC_ZLITO[4],                    ; SET CONDITION CODES
U 0566, 0980,0C37,0130,20A7,003F,9      ;10930         IRD1                            ; END OF INSTRUCTION
                        ;10931
                        ;10932         ;11-------------------------------; SRC > 0
                        ;10933         CC_ZLITO[0],                    ; SET CONDITION CODES
U 0567, 0180,0C37,0130,00A7,003F,9      ;10934         IRD1                            ;
```

```
                    ;10935 .TOC "  Floating point and CRC          : CMPF"
                    ;10936
                    ;10937 ;*****************************************************************************
                    ;10938 ;
                    ;10939 ;
                    ;10940 ;       51      CMPF    src1.rf,src2.rf
                    ;10941 ;
                    ;10942 ;
                    ;10943 ;               Input                   Q               src1.rf
                    ;10944 ;                                       MDR             src2.rf
                    ;10945 ;
                    ;10946 ;
                    ;10947 ;               Resources
                    ;10948 ;                                       Temp2           src1 from Q
                    ;10949 ;                                       Flag0           sign of src2
                    ;10950 ;                                        (SIGN(FLAG0))
                    ;10951 ;
                    ;10952 ;               Operation:
                    ;10953 ;
                    ;10954 ;                       Validate src2 (not reserved operand) set sign flag
                    ;10955 ;                       Get src1 to an Mbus source
                    ;10956 ;                       Validate src1 (not reserved operand)
                    ;10957 ;
                    ;10958 ;                       If src1 -       src2-src1 using ccop2
                    ;10959 ;                                       if src2 NOT neg then set N
                    ;10960 ;
                    ;10961 ;                       If src1 +       src1-src2 using ccop2
                    ;10962 ;                                       if src2 NEG then must clear both N and Z
                    ;10963 ;
                    ;10964 ;                               If src1 0       0-src2 using ccop1
                    ;10965 ;                                       setting condition codes on basis of src2
                    ;10966 ;
                    ;10967 ;       **********FPA ENTRY AND INTERFACE**********
                    ;10968 ;               (OS.RED) - FI.CMPF.REG/FI.CMPF.MEM
                    ;10969 ;
                    ;10970 ;*****************************************************************************
                    ;10971
                    ;10972 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                    ;10973 =0
                    ;10974 FP.CMPF:                                        ; **********************
                    ;10975         ;0-------------------------------;
                    ;10976         WB_EXP(M[MDR]),                         ; CHECK SRC2 FIRST FOR VALIDITY
                    ;10977         FRO.FLTZ?,PUSH,                         ;
U 00A6, 0C81,2437,0AB0,0047,0C57,0 323*  ;10978         NEXT/FP.CMPF.10                 ; PROCESS AS A SUBROUTINE
                    ;10979
                    ;10980         ;1-------------------------------;
                    ;10981         WB_EXP(M[TEMP2]),                       ; CHECK VALIDITY OF SRC1
                    ;10982         FRO.FLTZ?,                              ;
U 00A7, 0880,2437,0AB0,0047,0857,4 323*  ;10983         NEXT/FP.CMPF.30                 ;
```

```
                                ;10984 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;10985 =00
                                ;10986 FP.CMPF.10:
                                ;10987          ;00------------------------------; RESERVED OPERAND SRC2
U 0570, 0C80,0036,4030,0047,00FF,8  ;10988          NEXT/IE.OPER.FAULT              ;
                                ;10989
                                ;10990          ;01------------------------------; SRC2 < 0
                                ;10991          M[TEMP2] Q,                     ; COPY SRC1 INTO TEMP2
U 0571, 0C46,203A,40BD,8047,0000,1  ;10992          SET SIGN(FLAG0),RETURN [+1]     ; SET SIGN FLAG FOR ANSWER
                                ;10993
                                ;10994          ;10------------------------------; SRC2 = 0
                                ;10995          M[TEMP2] Q,                     ; COPY SRC1 INTO TEMP2
U 0572, 0486,203A,40BD,84E7,0000,1  ;10996          MDR_0,RETURN [+1]               ; COMPARISON AGAINST CLEAN ZERO
                                ;10997
                                ;10998 FX.MT2Q:                                 ; USED BY G H FLOAT
                                ;10999 FP.MT2Q:
                                ;11000          ;11------------------------------; SRC2 > 0
                                ;11001          M[TEMP2] Q,                     ; COPY SRC1 INTO TEMP2
U 0573, 0C86,203A,40BD,8047,0000,1  ;11002          RETURN [+1]                     ;
                                ;11003
                                ;11004 =00
                                ;11005 FP.CMPF.30:
                                ;11006          ;00------------------------------; RESERVED OPERAND SRC1
U 0574, 0C80,0036,4030,0047,00FF,8  ;11007          NEXT/IE.OPER.FAULT              ;
                                ;11008
                                ;11009          ;01------------------------------; SRC1 < 0
                                ;11010          WB_M[MDR]-R[TEMP2],             ; SRC2-SRC1
                                ;11011          CCOP2,SIZE[LONG],               ; SET CONDITION CODES
U 0575, 0881,2000,0420,9047,0055,8  ;11012          SIGN(FLAG0)?,NEXT/FP.CMPF.50    ; WAS SRC2 NEGATIVE
                                ;11013
                                ;11014          ;10------------------------------; SRC1 = 0
                                ;11015          WB_R[ZERO]-M[MDR],SIZE[LONG],   ; SRC1 IS 0
U 0576, 0481,2003,012D,8847,003F,9  ;11016          CCOP1,IRD1                      ; SET CONDITION CODES ON SRC2
                                ;11017
                                ;11018          ;11------------------------------; SRC1 > 0
                                ;11019          WB_R[TEMP2]-M[MDR],             ; SRC1-SRC2
                                ;11020          CCOP2,SIZE[LONG],               ; SET CONDITION CODES HERE
U 0577, 0881,2003,0420,9047,0050,4  ;11021          SIGN(FLAG0)?,NEXT/FP.CMPF.40    ; WAS SRC2 NEGATIVE
                                ;11022
                                ;11023 =0
                                ;11024 FP.CMPF.40:
                                ;11025          ;0-------------------------------; SRC1 & SRC2 ARE POSITIVE
U 0504, 0080,0036,4130,0047,003F,9  ;11026          IRD1                            ; CONDITION CODES SET CORRECTLY
                                ;11027
                                ;11028          ;1-------------------------------; SRC1 +, SRC2 -
U 0505, 0180,0C37,0130,00A7,003F,9  ;11029          CC_ZLITO[0],IRD1                ; CLEAR N & Z
                                ;11030
                                ;11031 =0
                                ;11032 FP.CMPF.50:
                                ;11033          ;0-------------------------------; SRC1 -, SRC2 +
U 0558, 0580,0C37,0130,40A7,003F,9  ;11034          CC_ZLITO[8],IRD1                ; SET N
                                ;11035
                                ;11036          ;1-------------------------------; SRC1 -, SRC2 -
U 0559, 0080,0036,4130,0047,003F,9  ;11037          IRD1                            ; CC SET CORRECTLY END INSTRUCTION
```

```
;11038 .TOC "  Floating point and CRC          : CMPD"
;11039
;11040 ;*****************************************************************************
;11041 ;
;11042 ;
;11043 ;       71      CMPD    src1.rd,src2.rd
;11044 ;
;11045 ;
;11046 ;               Input           Temp1 (bits 0-31)       src1.rd
;11047 ;                               MDR (bits 32-63)
;11048 ;
;11049 ;
;11050 ;               Resources       Temp2                   src1.rd
;11051 ;                               Temp3
;11052 ;
;11053 ;                               Temp1                   src2.rd
;11054 ;                               MDR
;11055 ;
;11056 ;                               Flag0                   sign of src1
;11057 ;                                 (SIGN(FLAG0))
;11058 ;
;11059 ;
;11060 ;               Subroutines:
;11061 ;                               OS.DRED                 get second operand
;11062 ;
;11063 ;
;11064 ;               Operation:
;11065 ;
;11066 ;                       Validate src1 (not reserved operand) and move
;11067 ;                       bits 0-31 to Temp2 and bits 32-63 to Temp3
;11068 ;
;11069 ;                       Get src2 in Temp1 and MDR
;11070 ;                       Validate src2 (not reserved operand)
;11071 ;
;11072 ;                       If src2 +  src1-src2 (bits 0-31) using ccop2
;11073 ;                                  if src1 neg then set N
;11074 ;                                  if both positive and equal must check
;11075 ;                                  second half
;11076 ;                                  by rotating each operand and setting
;11077 ;                                  condition codes using ccop1
;11078 ;
;11079 ;                       If src2 -  reverse the above operation
;11080 ;
;11081 ;
;11082 ;                       If src2 0  then 0-src2 (bits 0-31) using ccop2
;11083 ;                                  set condition codes on the basis of src1
;11084 ;
;11085 ;       **********FPA ENTRY AND INTERFACE**********
;11086 ;               (OS.FIDRED) - FI.CMPD.REG/FI.CMPF.MEM
;11087 ;
;11088 ;*****************************************************************************
```

```
                                ;11089 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;11090 FP.CMPD:                                 ; ********************
                                ;11091          ;-------------------------------;
                                ;11092          WB_EXP(M[TEMP1]),               ; CHECK SRC1 FOR VALIDITY
U 0331, 0880,1437,0AB0,0047,0857,8 323* ;11093  FRO.FLTZ?                       ;
                                ;11094
                                ;11095 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;11096 =00
                                ;11097          ;00-----------------------------; RESERVED OPERAND
U 0578, 0C80,0036,4030,0047,00FF,8      ;11098  NEXT/IE.OPER.FAULT              ;
                                ;11099
                                ;11100          ;01-----------------------------; SRC1 < 0
                                ;11101          R[TEMP2]_M[TEMP1],              ; MOVE BITS 0-31 TO TEMP2
                                ;11102          SET SIGN(FLAG0),                ; SET SIGN FLAG FOR ANSWER
U 0579, 0C44,1592,4030,8047,0051,3      ;11103  NEXT/FP.CMPD.10                 ;
                                ;11104
                                ;11105          ;10-----------------------------; SRC1 = 0
                                ;11106          R[TEMP2]_0,MDR_0,               ; ZERO OUT BOTH PARTS OF OPERAND
U 057A, 0084,05B7,0030,84E7,0051,3      ;11107  NEXT/FP.CMPD.10                 ;
                                ;11108
                                ;11109          ;11-----------------------------; SRC1 > 0
U 057B, 0C84,1592,4030,8047,0051,3      ;11110  R[TEMP2]_M[TEMP1]               ; MOVE BITS 0-31 TO TEMP2
                                ;11111
                                ;11112 FP.CMPD.10:
                                ;11113          ;-------------------------------;
                                ;11114          R[TEMP3]_M[MDR].RR.16,          ; SAVE SECOND HALF OF SRC1 ROTATED
                                ;11115          LOD INC DRA?,                   ; GET SRC2
U 0513, 0085,23B7,0190,C047,0018,0      ;11116  NEXT/OS.DREC                    ; RETURN IS THROUGH IRDROM FP.CMPD.20
                                ;11117
                                ;11118 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;11119 FP.CMPD.20:
                                ;11120          ;-------------------------------; ENTRY FROM IRDROM
                                ;11121          WB_EXP(M[TEMP1]),               ; CHECK SRC2 FOR VALIDITY
U 0335, 0080,1437,0A80,0047,0857,C 323* ;11122  FRO.FLTZ?                       ;
                                ;11123
                                ;11124 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;11125 =00      ;00-----------------------------; RESERVED OPERAND SRC2
U 057C, 0C80,0036,4030,0047,00FF,8      ;11126  NEXT/IE.OPER.FAULT              ;
                                ;11127
                                ;11128          ;01-----------------------------; SRC2 < 0
                                ;11129          Q_M[TEMP1]-R[TEMP2],            ; SRC2-SRC1
                                ;11130          CCOP2,SIZE[IDEP],               ; SET CONDITION CODES
                                ;11131          SIGN(FLAG0)?,                   ; WAS SRC1 NEGATIVE
U 057D, 0480,1000,1430,9047,0065,C      ;11132  NEXT/FP.CMPD.40                 ; GO FINISH THE COMPARISON
                                ;11133
                                ;11134          ;10-----------------------------; SRC2 = 0
                                ;11135          WB_M[TEMP2]-R[ZERO],SIZE[IDEP], ; USE SRC1 AS THE COMPARISON VALUE
U 057E, 0480,2000,013D,9047,003F,9      ;11136  CCOP2,IRD1                      ; SET CONDITION CODES
                                ;11137
                                ;11138          ;11-----------------------------; SRC2 > 0
                                ;11139          Q_M[TEMP2]-R[TEMP1],            ; SRC1-SRC2
                                ;11140          CCOP2,SIZE[IDEP],               ; SET CONDITION CODES
U 057F, 0C80,2000,1430,5047,0064,0      ;11141  SIGN(FLAG0)?,NEXT/FP.CMPD.30    ; WAS SRC1 NEGATIVE
```

```
                                ;11142 =0
                                ;11143 FP.CMPD.30:
                                ;11144          ;0------------------------------; BOTH SRC1 & SRC2 POSITIVE
                                ;11145          Q IS ZERO?,                     ; ARE THEY EQUAL IN THE FIRST PART
U 0640, 0080,003A,5A3D,8047,0067,4 ;11146       NEXT/FP.CMPD.50                 ; FINISH UP
                                ;11147
                                ;11148          ;1------------------------------; SRC1 -, SRC2 +
U 0641, 0580,0C37,0130,40A7,003F,9 ;11149       CC_ZLITO[8],IRD1                ; REVERSE THE CONDITION CODES
                                ;11150
                                ;11151 =0
                                ;11152 FP.CMPD.40:
                                ;11153          ;0------------------------------; SRC1 +,SRC2-
U 065C, 0180,0C37,0130,00A7,003F,9 ;11154       CC_ZLITO[0],IRD1                ; REVERSE CONDITION CODE SETTING
                                ;11155
                                ;11156          ;1------------------------------; SRC1 -,SRC2 -
                                ;11157          Q IS ZERO?,                     ; ARE THE FIRST PARTS EQUAL
U 065D, 0880,003A,5A3D,8047,0067,6 ;11158       NEXT/FP.CMPD.60                 ; FINISH UP
                                ;11159
                                ;11160 =0
                                ;11161 FP.CMPD.50:
                                ;11162          ;0------------------------------; FIRST PARTS OF SRC1 & SRC2 NOT EQUAL
U 0674, 0080,0036,4130,0047,003F,9 ;11163       IRD1                            ; END OF INSTRUCTION
                                ;11164
                                ;11165          ;1------------------------------; FIRST PARTS WERE EQUAL
U 0675, 0C81,23B7,0010,0467,0051,6 ;11166       MDR_M[MDR].RR.16                ; ROTATE SECOND PART OF SRC2
                                ;11167
                                ;11168          ;--------------------------------;
                                ;11169          WB_R[TEMP3]-M[MDR],SIZE[IDEP],  ; SRC1-SRC2
U 0516, 0081,2003,0130,C847,003F,9 ;11170       CCOP1,IRD1                      ; DO THE COMPARISON
                                ;11171 =0
                                ;11172 FP.CMPD.60:
                                ;11173          ;0------------------------------; FIRST PARTS OF SRC1 & SRC2 NOT EQUAL
U 0676, 0080,0036,4130,0047,003F,9 ;11174       IRD1                            ; END OF INSTRUCTION
                                ;11175
                                ;11176          ;1------------------------------; FIRST PARTS WERE EQUAL
U 0677, 0481,23B7,0010,0467,0051,8 ;11177       MDR_M[MDR].RR.16                ; ROTATE SECOND PART OF SRC2
                                ;11178
                                ;11179          ;--------------------------------;
                                ;11180          WB_M[MDR]-R[TEMP3],SIZE[IDEP],  ; SRC2-SRC1
U 0518, 0081,2000,0130,C847,003F,9 ;11181       CCOP1,IRD1                      ; SET CONDITION CODES
```

```
;11182 .TOC "  Floating point and CRC          : CVTFD"
;11183
;11184 ;*****************************************************************************
;11185 ;
;11186 ;
;11187 ;       56      CVTFD   src.rf,dst.wd
;11188 ;
;11189 ;
;11190 ;               Input           MDR             src.rf
;11191 ;
;11192 ;
;11193 ;               Resources       Temp2           bits 0-31
;11194 ;                               Temp3/MDR       bits 32-63 (0)
;11195 ;                               Q               to hold operand for writing
;11196 ;                               Flag1           for writing bits 32-63
;11197 ;                                 (WRITE(FLAG1))
;11198 ;
;11199 ;               Subroutine:
;11200 ;                               OS.WRT1         get  dst.wd
;11201 ;
;11202 ;               Exit
;11203 ;
;11204 ;                               FP.WRITE.SEC    to write second half of  dst.wd
;11205 ;
;11206 ;               Operation:
;11207 ;
;11208 ;                       Validate floating point operand (not reserved)
;11209 ;                       If register mode then write out bits 0-31 from
;11210 ;                       Temp2 set flag to write from MDR
;11211 ;                       and exit to FP.WRITE.SEC
;11212 ;                       If not register mode clear flag and set Temp3 to 0
;11213 ;                       Call OS.WRT1 to get dst.wd and return to
;11214 ;                       exit to FP.WRITE.SEC
;11215 ;       **********FPA ENTRY AND INTERFACE**********
;11216 ;               (OS.RED) - FI.ADDD3.MEM
;11217 ;
;11218 ;*****************************************************************************
```

```
                                        ;11219 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;11220 =1111
                                        ;11221 FP.CVTFD:                                ; **********************
                                        ;11222          ;1111-----------------------------;
                                        ;11223          SIZE[LONG],                     ; NULLIFY THE ZEXT
                                        ;11224          WB_EXP(M[MDR]) Q_ZEXT(MB),      ; TEST VALIDITY OF SRC
                                        ;11225          SET WRITE(FLAG1),               ; SET FLAG1 FOR LATER WRITE TEST
U 001F, 0C49,241C,7AA0,0047,0858,0 323* ;11226          FRO.FLTZ?                       ; SAVING VALUE IN Q
                                        ;11227
                                        ;11228 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                        ;11229 =00      ;00-----------------------------; RESERVED OPERAND
U 0580, 0C80,0036,4030,0047,00FF,8      ;11230          NEXT/IE.OPER.FAULT              ;
                                        ;11231
                                        ;11232          ;01-----------------------------; SRC < 0
                                        ;11233          M[TEMP2] Q,MDR_0,               ; FIRST HALF IN TEMP2 ZERO SECOND HALF
                                        ;11234          REG MODE?,                      ; WAS IT REGISTER MODE
U 0581, 0886,203A,493D,84E7,000A,C      ;11235          NEXT/FP.CVTFD.20                ; FINISH UP
                                        ;11236
                                        ;11237          ;10-----------------------------; SRC = 0
                                        ;11238          R[TEMP2] 0, MDR_0,              ; ZERO OUT OPERAND
                                        ;11239          REG MODE?,                      ; WAS IT REGISTER MODE
U 0582, 0884,05B7,0930,84E7,000A,C      ;11240          NEXT/FP.CVTFD.20                ; FINISH UP
                                        ;11241
                                        ;11242          ;11-----------------------------; SRC > 0
                                        ;11243          M[TEMP2] Q, MDR_0,              ; FIRST HALF TO TEMP2 ZERO SECOND HALF
                                        ;11244          REG MODE?,                      ; WAS IT REGISTER MODE
U 0583, 0886,203A,493D,84E7,000A,C      ;11245          NEXT/FP.CVTFD.20                ; FINISH UP
                                        ;11246 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;11247 =0
                                        ;11248 FP.CVTFD.20:
                                        ;11249          ;0-----------------------------; NOT REGISTER MODE
                                        ;11250          R[TEMP3] 0,                     ; ZERO OUT SECOND HALF
                                        ;11251          CLEAR WRITE(FLAG1),             ; NOW WILL WRITE TEMP3 NOT MDR
                                        ;11252          BRA ON ADD?,                    ; GET DST ADDRESS
U 00AC, 040C,05B7,0630,C047,0014,0      ;11253          NEXT/OS.WRT1                    ; RETURN IS THROUGH IRDROM
                                        ;11254                                          ; TO FP.CVTFD.30
                                        ;11255
                                        ;11256 FP.CVTFD.30:
                                        ;11257          ;1-----------------------------; REGISTER MODE
                                        ;11258          R[DST.R] M[TEMP2],              ; WRITE OUT THE FIRST HALF
                                        ;11259          WRITE NOTREG,SIZE[IDEP],        ;
                                        ;11260          SET MM.NOINT,                   ; NO INTERRUPTS ALLOWED
                                        ;11261          CCOP1,                          ; SET CONDITION CODES
U 00AD, 0064,2592,403C,4DDA,0056,F      ;11262          NEXT/FP.WRITE.SEC               ;
```

```
;11263 .TOC "  Floating point and CRC          : CVTBF CVTWF CVTBD CVTWD CVTLD & CVTLF"
;11264
;11265 ;*****************************************************************************
;11266 ;
;11267 ;       4C      CVTBF   src.rb,dst.wf
;11268 ;       4D      CVTWF   src.rw,dst.wf
;11269 ;       6C      CVTBD   src.rb,dst.wd
;11270 ;       6D      CVTWD   src.rw,dst.wd
;11271 ;       6E      CVTLD   src.rl,dst.wd
;11272 ;
;11273 ;               Input:          MDR             src.rx
;11274 ;               Resources:
;11275 ;                               Temp2           src fraction being formed
;11276 ;                               Temp5           exponent
;11277 ;                               Flag0           sign of operand
;11278 ;                                 (SIGN(FLAG0))
;11279 ;                               Flag1           write out flag
;11280 ;                                 (WRITE(FLAG1))
;11281 ;
;11282 ;               Subroutines:
;11283 ;                               OS.WRT1         to get dst.wx
;11284 ;
;11285 ;               Exit:
;11286 ;                               FP.PACK.T2T5    to pack write out operand
;11287 ;                               FP.WRITE.FIRST  to write out zero answer
;11288 ;               Operation:
;11289 ;                       Sign extend src according to instruction size
;11290 ;                       Get dst.wx operand (OS.WRT1)
;11291 ;                       Complements value if negative then or otherwise
;11292 ;                       does a find first set  and extracts the fraction
;11293 ;                       removing the hidden bit and forming the exponent
;11294 ;                       from the position contained in platch and biasing it
;11295 ;                       Exit is to FP.PACK.T2T5 to pack and write value
;11296 ;
;11297 ;       4E      CVTLF   src.rd,dst.wf
;11298 ;
;11299 ;               Input           Q               src.rl
;11300 ;                               VA/RNUM         dst.wf
;11301 ;               Resources:
;11302 ;                               Same as for other instructions abouve
;11303 ;               Exit:
;11304 ;                               Same as above
;11305 ;               Operation:
;11306 ;                       After performing the same operation as above
;11307 ;                       the fraction is rounded and the exponent incremented
;11308 ;                       if appropriate  exit is through common flows
;11309 ;
;11310 ;         **********FPA ENTRY AND INTERFACE**********
;11311 ;     FOR CVTBF,CVTWF,CVTLF               (OS.RED) - FI.ADDF3.MEM
;11312 ;     FOR CVTBD,CVTWD,CVTLD               (OS.RED) - FI.ADDD3.MEM
;11313 ;
;11314 ;*****************************************************************************
```

```
                                        ;11315 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;11316 =0
                                        ;11317 FP.CVT.BWL.FD:                          ; **********************
                                        ;11318         ;0-------------------------------;
                                        ;11319         R[TEMP2]_SEXT(M[MDR]),          ; SIGN EXTEND THE SRC ACCORDING TO SIZE
                                        ;11320         PUSH,SIZE[IDEP],                ; RETURN HERE FOR FINAL WORK
                                        ;11321         LOD INC BRA?,                   ; GET DESTINATION ADDRESS
U 00B2, 0885,2E5E,01B0,8047,0414,0      ;11322         NEXT/OS.WRT1                    ; RET  THRU IRDROM TO FP.CVTBWL.FD.100
                                        ;11323
                                        ;11324 FP.CVT.BWL.FD.300:
                                        ;11325         ;1-------------------------------;
                                        ;11326         M[TEMP5]_MB+ZLIT0[81],          ; ADD BIAS AND FACTOR OF 1 TO EXPONENT
                                        ;11327         SET WRITE(FLAG1),               ; SET FLAG1 FOR LATER DOUBLE WRITE TEST
                                        ;11328         SIGN(FLAG0)?,                   ; WAS THE SIGN NEGATIVE
U 00B3, 054E,5C11,0434,0847,0068,8      ;11329         NEXT/FP.PACK.T2T5               ; PACK AND WRITE OPERAND
                                        ;11330
                                        ;11331 =0
                                        ;11332 FP.CVTLF:                               ; **********************
                                        ;11333         ;0-------------------------------;
                                        ;11334         M[TEMP2]_Q,                     ; SRC IN Q
                                        ;11335         PUSH,SIZE[IDEP],                ; RETURN HERE FROM PROCESSING
                                        ;11336         SIGND CMP?,                     ; WHAT WAS SIGN OF OPERAND
U 00BC, 0886,203A,4B7D,8047,0458,4      ;11337         NEXT/FP.CVT.BWL.FD.200          ; DO THE TESTING
                                        ;11338
                                        ;11339         ;1-------------------------------;
U 00BD, 0986,2C11,0038,0047,0052,7      ;11340         M[TEMP2]_MB+ZLIT0[100]          ; ROUND THE FRACTION HIDDEN BIT GONE
                                        ;11341
                                        ;11342 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                        ;11343         ;-------------------------------;
                                        ;11344         M[TEMP5]_MB+R[ZERO]+ALKC,       ; ADD THE CARRY IF THERE
U 0527, 0886,5041,003D,8047,000B,3      ;11345         NEXT/FP.CVT.BWL.FD.300          ; JOIN COMMON END UP
                                        ;11346
                                        ;11347
                                        ;11348 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;11349 FP.CVT.BWL.FD.100:
                                        ;11350         ;-------------------------------;
                                        ;11351         M[TEMP2]_MB+R[ZERO],SIZE[IDEP], ; GET THE SIGN AGAIN
                                        ;11352         SIGND CMP?,MDR_0,               ; ZERO OUT SECOND HALF IN CASE DOUBLE
U 0339, 0086,2001,0B7D,84E7,0858,4 380* ;11353         NEXT/FP.CVT.BWL.FD.200          ; DO MAIN PROCESSING
```

```
                                ;11354 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;11355 =00
                                ;11356 FP.CVT.BWL.FD.200:
                                ;11357          ;00-------------------------------; VALUE > 0
                                ;11358          PL_MSS M[TEMP2],                  ; FIND FIRST 1 BIT
U 0584, 0480,29C2,403D,8047,0058,7  ;11359      NEXT/FP.CVT.BWL.FD.220            ; CONTINUE WITH PROCESSING
                                ;11360
                                ;11361          ;01-------------------------------; VALUE = 0
                                ;11362          IR<5>?                            ; IS THIS A DOUBLE OR FLOAT  WRITE
                                ;11363          SET WRITE(FLAG1),                 ; SET FLAG FOR TEST
U 0585, 0048,0036,48F0,0047,0050,8  ;11364      NEXT/FP.WRITE.FIRST               ;
                                ;11365
                                ;11366          ;10-------------------------------; VALUE < 0
                                ;11367          M[TEMP2]_-MB,                     ; 2S COMPLEMENT THE VALUE
                                ;11368          SET SIGN(FLAG0),                  ; SRC IS NEGATIVE
U 0586, 0446,2003,003D,8047,0058,4  ;11369      NEXT/FP.CVT.BWL.FD.200            ; NOW FIND FIRST SET
                                ;11370
                                ;11371 FP.CVT.BWL.FD.220:
                                ;11372          ;-------------------------------;
                                ;11373          M[TEMP2]_(MB R[ZERO]).RR.P,       ; EXTRACT THE 32 BITS OF FRACTION
U 0587, 0086,2137,08FD,8047,0067,8  ;11374      IR<5>?                            ; IS IT CVTBD,CVTWD,CVTLD
                                ;11375
                                ;11376 =0
                                ;11377 FP.CVT.BWL.FD.250:
                                ;11378          ;0--------------------------------; NOT DOUBLE
                                ;11379          M[TEMP5]_PL,                      ; SAVE PLATCH FOR COMPUTING  EXPONENT
U 0678, 0C86,5B37,00B0,0047,0000,1  ;11380      RETURN [+1]                       ; RETURN TO CALLING ROUTINE
                                ;11381
                                ;11382          ;1--------------------------------; DOUBLE
U 0679, 0C80,22F7,003D,8467,0053,D  ;11383      MDR_(M[TEMP2] R[ZERO]).RR.9       ; SAVE LOW BITS IN SECOND HALF
                                ;11384
                                ;11385          ;-------------------------------;
                                ;11386          MDR_M[MDR].RR.16,                 ; SAVE LOW BITS IN SECOND HALF
U 053D, 0481,23B7,0010,0467,0067,8  ;11387      NEXT/FP.CVT.BWL.FD.250            ;
```

```
;11388 .TOC "  Floating point and CRC          : CVTFB CVTFW CVTFL"
;11389
;11390 ;*****************************************************************************
;11391 ;
;11392 ;       48      CVTFB   src.rf,dst.wb
;11393 ;       49      CVTFW   src.rf,dst.ww
;11394 ;       4A      CVTFL   src.rf,dst.wl
;11395 ;
;11396 ;
;11397 ;               Input           Q               src.rf
;11398 ;                               VA/RNUM         dst.wx
;11399 ;
;11400 ;
;11401 ;               Resources:
;11402 ;                               Temp2           fraction
;11403 ;                               Temp3           zero
;11404 ;                               Temp4           for size test
;11405 ;                               Temp5           exponent
;11406 ;                               MDR             integer
;11407 ;                               Flag0           sign of operand
;11408 ;                                  (ADD1(FLAG0))
;11409 ;                               Flag2           overflow flag
;11410 ;                                  (OVER(FLAG2))
;11411 ;
;11412 ;               Subroutines:
;11413 ;                               FP.ADDFOP1      parse operand
;11414 ;                               FP.CVTFI.SUB    convert float to  integer
;11415 ;
;11416 ;
;11417 ;                       Operation:
;11418 ;                               Entry after src and dst evaluated
;11419 ;                               Parse and validate operand (FP.ADDFOP1)
;11420 ;                               Shift off overflow bit
;11421 ;                               Call FP.CVTFI.SUB to obtain integer
;11422 ;                               Integer returned in MDR
;11423 ;                               Integer sign extended   according to instruction size
;11424 ;                               **if positive
;11425 ;                                       it  is tested for integer overflow into dst size
;11426 ;                               **if negative
;11427 ;                                       the value is complemented
;11428 ;                               and the test performed as above
;11429 ;                               In either case the operand is written
;11430 ;                               If integer overflow has been detected the V bit is set
;11431 ;
;11432 ;               **********FPA ENTRY AND INTERFACE**********
;11433 ;                       (OS.RED) - FI.ADDF3.MEM
;11434 ;
;11435 ;*****************************************************************************
```

```
                                ;11436 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;11437 =0
                                ;11438 FP.CVTF.BWL:                             ; **********
                                ;11439         ;0-------------------------------;
                                ;11440         M[TEMP2] 0,                      ; MOVE SRC1 TO AN MBUS SOURCE
U 00C6, 0086,203A,403D,8047,0467,A ;11441      PUSH,NEXT/FP.CVTF.100            ; COMMON FLOWS RET AFTER FP.CVTFI.SUB
                                ;11442
                                ;11443 FP.CVTF.200:
                                ;11444         ;1-------------------------------;
                                ;11445         R[TEMP4] SEXT(M[MDR]),           ; SIGN EXTEND THE INTEGER FOR
                                ;11446         SIZE[IDEP],                      ; TEST IF FITS INTO SIZE OF DST
                                ;11447         ADD1(FLAG0)?,                    ; POSITIVE OR NEG INTEGER
U 00C7, 0C85,2E5E,0431,0047,0067,C ;11448      NEXT/FP.CVTF.250                 ; CONTINUE TEST
                                ;11449
                                ;11450 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;11451 =0
                                ;11452 FP.CVTF.100:
                                ;11453         ;0-------------------------------;
                                ;11454         R[TEMP3] 0,                      ; ZERO THE OTHER PART FOR LATER
U 067A, 0484,05B7,0030,C047,045A,1 ;11455      PUSH,NEXT/FP.ADDFOP1             ; PARSE SRC1
                                ;11456
                                ;11457         ;1-------------------------------;
                                ;11458         R[TEMP2] RB.ASL.1,               ; SHIFT LEFT 1 FOR FP.CVTFI.SUB
U 067B, 0884,05F7,0010,8047,006E,A ;11459      NEXT/FP.CVTFI.SUB                ; CONVERT TO INTEGER RET TO CALLING RTN
                                ;11460
                                ;11461 =0
                                ;11462 FP.CVTF.250:
                                ;11463         ;0-------------------------------;
                                ;11464         WB_M[MDR]-R[TEMP4],              ; INTEGER IS POSITIVE
                                ;11465         SIZE[IDEP],                      ; CF  FOR INTEGER OVERFLOW FOR DST SIZE
                                ;11466         WX.EQ.0?,                        ; WILL IT FIT
U 067C, 0C81,2000,0A31,0047,0867,E 345* ;11467 NEXT/FP.CVTF.300                 ;
                                ;11468
                                ;11469         ;1-------------------------------; INTEGER IS NEGATIVE
                                ;11470         MDR -M[MDR],                     ; 2S COMPLEMENT THE INTEGER
                                ;11471         SIZE[IDEP],                      ;
                                ;11472         CLEAR ADD1(FLAG0),               ; CLEAR THE SIGN FLAG NOW
U 067D, 0001,2003,003D,8467,000C,7 ;11473      NEXT/FP.CVTF.200                 ; GO BACK AND DO TEST
                                ;11474
                                ;11475 =0
                                ;11476 FP.CVTF.300:
                                ;11477         ;0-------------------------------; WRITE KNOWN INTEGER OVERFLOW CASE
                                ;11478         R[DST.R].SIZ_M[MDR],             ; INTEGER IN MDR
                                ;11479         WRITE NOTREG,SIZE[IDEP],         ;
                                ;11480         SET MM.NOINT,                    ; NO INTERRUPTS TILL END
                                ;11481         CCOP1,                           ; SET CONDITION CODES
U 067E, 0463,2592,403C,4DDA,0051,E ;11482      NEXT/FP.CVTF.450                 ; GO SET V AND END
                                ;11483
                                ;11484         ;1-------------------------------; WRITE OVERFLAG MAY BE SET
                                ;11485         R[DST.R].SIZ_M[MDR],             ; INTEGER IN MDR
                                ;11486         WRITE NOTREG,SIZE[IDEP],         ;
                                ;11487         SET MM.NOINT,                    ; NO INTERRUPTS TILL END
                                ;11488         CCOP1,                           ; SET CONDITION CODES
U 067F, 0C63,2592,44BC,4DDA,0051,A ;11489      OVER(FLAG2)?                     ; INTEGER OVERFLOW PREVIOUSLY DETECTED
```

```
                                ;11490 =0**
                                ;11491 FP.CVTF.400:
                                ;11492          ;0**-------------------------------;
U 051A, 0080,0036,4130,0047,003F,9 ;11493       IRD1                                ; END OF INSTRUCTION A WASTE
                                ;11494
                                ;11495 FP.SETVIRD1:
                                ;11496 FP.CVTF.450:
                                ;11497          ;1**-------------------------------;
                                ;11498          SET V,                              ; SET INTEGER OVERFLOW IN PSL
U 051E, 0480,0036,4130,08E7,003F,9 ;11499       IRD1                                ; END OF INSTRUCTION
```

```
;11500 .TOC "  Floating point and CRC          : CVTDB CVTDW CVTDL"
;11501
;11502 ;*****************************************************************************
;11503 ;
;11504 ;       68      CVTDB   src.rd,dst.wb
;11505 ;       69      CVTDW   src.rd,dst.ww
;11506 ;       6A      CVTDL   src.rd,dst.wl
;11507 ;
;11508 ;
;11509 ;               Input           Temp1                   src.rd (bits 0-31)
;11510 ;                               MDR                     (bits 32-63)
;11511 ;               Resources:
;11512 ;                               Temp2                   fraction (bits 0-31)
;11513 ;                               Temp3                   ( bits 32-63)
;11514 ;                               Temp4                   size test register
;11515 ;                               Temp5                   exponent
;11516 ;                               MDR                     integer
;11517 ;                               Q                       fraction
;11518 ;                               Flag0                   sign of operand
;11519 ;                                   (ADD1(FLAG0))
;11520 ;                               Flag2                   overflow flag
;11521 ;                                   (OVER(FLAG2))
;11522 ;
;11523 ;               Subroutines:
;11524 ;                               FP.ADDDOP1              parse operand
;11525 ;                               OS.WRT1                 get dst.wx
;11526 ;                               FP.CVTFI.SUB            convert float to integer
;11527 ;
;11528 ;               Exit:
;11529 ;                               FP.CVTF.250             common flow finish up
;11530 ;
;11531 ;               Operation:
;11532 ;                       Parse operand via FP.ADDDOP1
;11533 ;                       Shift operand left 1 to remove overflow bit
;11534 ;                       Call OS.WRT1 to get dst.wx and return through
;11535 ;                       IRDROM to FP.CVTD.250
;11536 ;                       Call FP.CVTFI.SUB to do conversion
;11537 ;                       Start test to see if integer in MDR fits into
;11538 ;                       dst according to instruction size
;11539 ;                       Exit to FP.CVTF.250 to join common flow with
;11540 ;                       float to integer single precision
;11541 ;
;11542 ;       **********FPA ENTRY AND INTERFACE**********
;11543 ;               (OS.FIDRED) - FI.ADDF3.MEM
;11544 ;
;11545 ;*****************************************************************************
```

```
                                        ;11546 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;11547 =00
                                        ;11548 FP.CVTD.BWL:                          ; **********************
                                        ;11549         ;00-----------------------------;
                                        ;11550         R[TEMP5]_EXP(M[TEMP1]),       ; PARSE FIRST OPERAND
                                        ;11551         FRO.FLTZ?,                    ; TEST VALIDITY
U 0068, 0C84,1437,0AB1,4047,0C5C,4 323* ;11552         PUSH,NEXT/FP.ADDDOP1          ;
                                        ;11553
                                        ;11554         ;01-----------------------------;
                                        ;11555         M[TEMP3]_(MB Q).RL.1,         ; Q HAS BITS0-31 FROM FP.ADDDOP1
                                        ;11556         PUSH,                         ; NEXT OPERAND RETURN THRU IRDROM TO
                                        ;11557         LOD INC BRA?,                 ; FP.CVTD.150
U 0069, 0886,3305,C1BD,8047,0414,0      ;11558         NEXT/OS.WRT1                  ; RETURN HERE AFTER HAVE INTEGER
                                        ;11559
                                        ;11560         ;10-----------------------------;
                                        ;11561         R[TEMP4]_SEXT(M[MDR]),        ; TESTING NOW TO SEE IF FITS INTO SIZE
                                        ;11562         SIZE[IDEP],                   ; OF DST
                                        ;11563         ADD1(FLAG0)?,                 ; WAS SRC1 POSITIVE
U 006A, 0C85,2E5E,0431,0047,0067,C      ;11564         NEXT/FP.CVTF.250              ; CONTINUE PROCESSING
                                        ;11565
                                        ;11566 =
                                        ;11567
                                        ;11568 FP.CVTD.150:
                                        ;11569         ;-----------------------------; ENTER HERE THRU IRDDOM FOR CVTD'S
                                        ;11570         M[TEMP2]_Q,                   ; BITS 0-31 FROM 1 TO TEMP2
U 033D, 0886,203A,403D,8047,006E,A      ;11571         NEXT/FP.CVTFI.SUB             ; GET INTEGER
```

```
;11572 .TOC "  Floating point and CRC          : CVTRFL CVTRDL"
;11573
;11574 ;*****************************************************************************
;11575 ;
;11576 ;       4B      CVTRFL  src.rf,dst.wl
;11577 ;
;11578 ;               Input           Q               src.rf
;11579 ;                               VA/RNUM         dst.wl
;11580 ;
;11581 ;               Resources:
;11582 ;                               Same as CVTFD,CVTFW,CVTFL
;11583 ;               Exit:
;11584 ;                               FP.CVTRDL.30
;11585 ;               Operation:
;11586 ;                       Common flows with other float (single precision)
;11587 ;                       to integer routines
;11588 ;                       However, after return from computing integer
;11589 ;                       shift left to get carry  join common flow with CVTRDL
;11590 ;
;11591 ;       6B      CVTRDL  src.rd,dst.wl
;11592 ;
;11593 ;               Input           Temp1                   src.rd (bits 0-31)
;11594 ;                               MDR                     (bits 32-63)
;11595 ;               Resources:
;11596 ;                               Flag2 to indicate round in FP.CVTFI
;11597 ;                               Same as for CVTDB,CVTDW,CVTDL
;11598 ;               Operation:
;11599 ;                       Same operation as  the above double precision
;11600 ;                       After computing integer return
;11601 ;                       (Temp2 and Temp3 have fraction position so that
;11602 ;                       binary point is before bit 31 of Temp2)
;11603 ;                       Shift round bit into carry
;11604 ;                       Test for integer overflow as detected by CVTFI.SUB
;11605 ;                       If not  then carry (or no carry ) is added to integer
;11606 ;                       and test again for integer overflow
;11607 ;                       also test for -2**32 case
;11608 ;                       i.e. if integer value was negative after carry added
;11609 ;                       and sign indicated value not negative
;11610 ;                       this is integer overflow write  low bits set V and exit
;11611 ;
;11612 ;                       if integer overflow has been previously detected
;11613 ;                       or if this is CVTRFL instruction
;11614 ;                       where no integer overflow is possible then bit 31 is
;11615 ;                       irrelevant  Add carry to integer
;11616 ;                       and complement it if negative was expected
;11617 ;                       if integer overflow flag set then set V and exit
;11618 ;                       otherwise just exit
;11619 ;
;11620 ;       **********FPA ENTRY AND INTERFACE**********
;11621 ;       FOR CVTRFL      (OS.RED)    - FI.ADDF3.MEM
;11622 ;       FOR CVTRDL      (OS.FIDRED) - FI.ADDD3.MEM
;11623 ;
;11624 ;*****************************************************************************
```

```
                                ;11625 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;11626 =0
                                ;11627 FP.CVTRFL:                                ; **********************
                                ;11628         ;0-------------------------------;
                                ;11629         M[TEMP2]_Q,SET FLAG2,             ; GET SRC1 TO AN MBUS
U 00CC, 0856,203A,403D,8047,0467,A ;11630      PUSH,NEXT/FP.CVTF.100             ; RETURN HERE WHEN HAVE INTEGER
                                ;11631
                                ;11632         ;1-------------------------------;
                                ;11633         R[TEMP2]_M[TEMP2]+RB,             ; SHIFT LEFT 1 BIT INTO CARRY
U 00CD, 0084,2001,0030,8047,005A,F ;11634      NEXT/FP.CVTRDL.30                 ; THIS IS ROUND BIT
                                ;11635
                                ;11636 =00
                                ;11637 FP.CVTRDL:                                ; **********************
                                ;11638         ;00-------------------------------;
                                ;11639         R[TEMP5]_EXP(M[TEMP1]),           ; PARSE SRC1
                                ;11640         FRO.FLTZ?,                        ; TEST VALIDITY
U 0074, 0C84,1437,0AB1,4047,0C5C,4 323* ;11641 PUSH,NEXT/FP.ADDDOP1              ;
                                ;11642
                                ;11643         ;01------------------------------; RET THRU IRDROM TO FP.CVTD.150
                                ;11644         M[TEMP3]_(MB Q).RL.1,             ; Q HAS BITS 0-31 NOW SHIFTING LEFT 1
                                ;11645         SET FLAG2,                        ; USE TO INDICATE ROUND IN FP.CVTFI
                                ;11646         PUSH,LOD INC BRA?,                ; RET HERE AFTER HAVE INTEGER
U 0075, 0056,3305,C1BD,8047,0414,0 ;11647      NEXT/OS.WRT1                      ; FOR ROUNDING
                                ;11648
                                ;11649         ;10------------------------------;
                                ;11650         R[TEMP2]_M[TEMP2]+RB,             ; SHIFT ROUND BIT OF FRACTION INTO CARRY
U 0076, 0884,2001,0480,8047,005A,B ;11651      OVER(FLAG2)?                      ; WAS INTEGER OVERFLOW ALREADY DETECTED
                                ;11652 =
                                ;11653 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;11654 =0**
                                ;11655         ;0**-----------------------------  NOT YET MDR + EXCEPT -2**32
                                ;11656         MDR_M[MDR]+ALKC,                  ; ADD CARRY TO INTEGER
                                ;11657         SIZE[LONG],WB<31-30>?,            ; CHECK ON BIT 31
U 05AB, 0481,2041,06ED,8467,084F,9 344* ;11658 NEXT/FP.CVTRDL.40                 ; FURTHER PROCESSING
                                ;11659
                                ;11660 FP.CVTRDL.30:
                                ;11661         ;1**-----------------------------; ENTRY  CVTF.RFL INTEGER OVERFLOW  RDL
                                ;11662         MDR_M[MDR]+ALKC,                  ; CANT OVERFLOW IF RFL & IF OVERFLOWED
U 05AF, 0C81,2041,043D,8467,0068,0 ;11663      ADDT(FLAG0)?                      ; FOR RDL SIGN OF CARRY OUT  IRRELEVANT
                                ;11664
                                ;11665 =0
                                ;11666         ;0-------------------------------; INTEGER POSITIVE
                                ;11667         R[DST.R]_SIZ_M[MDR],              ; INTEGER IN MDR
                                ;11668         WRITE NOTREG,SIZE[IDEP],          ;
                                ;11669         SET MM.NOINT,                     ; NO INTERRUPTS TILL END
                                ;11670         CCOP1,                            ; SET CONDITION CODES
                                ;11671         OVER(FLAG2)?,                     ; WAS INTEGER OVERFLOW SET
U 0680, 0C63,2592,44BC,4DDA,0051,A ;11672      NEXT/FP.CVTF.400                  ; SET V AND OR END
                                ;11673
                                ;11674         ;1-------------------------------; INTEGER IS NEGATIVE
                                ;11675         R[DST.R]_SIZ_-M[MDR],             ; INTEGER IN MDR COMPLEMENT IT
                                ;11676         WRITE NOTREG,SIZE[IDEP],          ;
                                ;11677         SET MM.NOINT,CCOP1,               ; NO INTERRUPTS TILL END
  0681, 0C63,2593,04BC,4DDA,0051,A ;11678      OVER(FLAG2)?,NEXT/FP.CVTF.400     ; INTEGER OVERFLOW DETECTED BEFORE
```

```
                                ;11679 =01
                                ;11680 FP.CVTRDL.40:
                                ;11681          ;01-----------------------------; VALUE WAS POSITIVE IN MDR AFTER ALKC
                                ;11682          ADD1(FLAG0)?,                   ; WHAT WAS THE SIGN ORIGINALLY
U 04F9, 0080,0036,4430,0047,0068,2 ;11683       NEXT/FP.CVTRDL.50               ; CONTINUE PROCESSING
                                ;11684
                                ;11685          ;11-----------------------------; VALUE WAS NEGATIVE IN MDR AFTER ALKC
                                ;11686          R[DST.R].SIZ_-M[MDR],           ; INTEGER IN MDR COMPLEMENT IT
                                ;11687          WRITE NOTREG,SIZE[IDEP],        ;
                                ;11688          SET MM.NOINT,                   ; NO INTERRUPTS TILL END
                                ;11689          CCOP1,                          ; SET CONDITION CODES
                                ;11690          ADD1(FLAG0)?,                   ; WAS THE SIGN CORRECT
U 04FB, 0063,2593,043C,4DDA,0068,4 ;11691       NEXT/FP.CVTRDL.60               ; HERE FILTERING CASE OF -2**32
                                ;11692
                                ;11693 =0
                                ;11694 FP.CVTRDL.50:
                                ;11695          ;0-----------------------------; POSITIVE SIGN FLAG AFTER ROUND
                                ;11696          R[DST.R].SIZ_M[MDR],            ; INTEGER IN MDR
                                ;11697          WRITE NOTREG,SIZE[IDEP],        ;
                                ;11698          CCOP1,                          ; SET CONDITION CODES
U 0682, 0883,2592,413C,4DDA,003F,9 ;11699       IRD1                            ; END OF INSTRUCTION
                                ;11700
                                ;11701
                                ;11702          ;1-----------------------------; NEGATIVE SIGN FLAG AFTER ROUND
                                ;11703          R[DST.R].SIZ_-M[MDR],           ; INTEGER IN MDR
                                ;11704          WRITE NOTREG,SIZE[IDEP],        ; COMPLEMENT VALUE
                                ;11705          CCOP1,                          ; SET CONDITION CODES
U 0683, 0883,2593,013C,4DDA,003F,9 ;11706       IRD1                            ; END OF INSTRUCTION
                                ;11707
                                ;11708 =0
                                ;11709 FP.CVTRDL.60:
                                ;11710          ;0-----------------------------; INTEGER OVERFLOW FROM ROUNDING
                                ;11711          SET V,                          ;
U 0684, 0480,0036,4130,08E7,003F,9 ;11712       IRD1                            ;
                                ;11713
                                ;11714          ;1-----------------------------; NO INTEGER OVERFLOW FROM ROUNDING
U 0685, 0080,0036,4130,0047,003F,9 ;11715       IRD1                            ;
```

```
;11716 .TOC "  Floating point and CRC          : CVTDF"
;11717
;11718 ;*****************************************************************************
;11719 ;
;11720 ;
;11721 ;       76      CVTDF   src.rd,dst.wf
;11722 ;
;11723 ;               Input           Temp1                   src.rd (bits 0-31)
;11724 ;                               MDR                     (bits 32-63)
;11725 ;
;11726 ;               Resources:
;11727 ;                               Temp1                   fraction (bits 0-31)
;11728 ;                               Temp2                   fraction (bits 0-31)
;11729 ;                               Temp5                   exponent
;11730 ;                               Flag0                   sign of operand
;11731 ;                                 (SIGN(FLAG0))
;11732 ;
;11733 ;
;11734 ;               Operation:
;11735 ;                       Test validity of operand (not reserved operand)
;11736 ;                       If valid test the round bit
;11737 ;                       If round bit not set then write out packed operand
;11738 ;                       If round set then unpack fraction bits 0-31 only
;11739 ;                       add 1 to round position
;11740 ;                       if no carry then shift left to remove hidden bit and
;11741 ;                       overflow bit and pack and write
;11742 ;
;11743 ;                       if carry  increment exponent  test for floating overflow
;11744 ;                       if none  shift to remove hidden and overflow bits
;11745 ;                       and exit as above
;11746 ;                       If floating overflow fault to IE.FLOW.FAULT
;11747 ;
;11748 ;
;11749 ;       **********FPA ENTRY AND INTERFACE**********
;11750 ;               (OS.FIDRED) - FI.ADDF3.MEM
;11751 ;
;11752 ;*****************************************************************************
```

```
                                        ;11753 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;11754 FP.CVTDF:                        ; **********************
                                        ;11755          ;-----------------------;
U 0341, 0C80,003A,403D,8467,0054,6      ;11756          MDR_Q                   ; OS.WRT2 MOVED MDR(BITS32-63)  TO Q
                                        ;11757
                                        ;11758 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                        ;11759          ;-----------------------;
                                        ;11760          R[TEMP5]_EXP(M[TEMP1]),  ; TEST VALIDITY OF OPERAND
U 0546, 0C84,1437,0AB1,4047,0858,8 323* ;11761          FRO.FLTZ?               ;
                                        ;11762
                                        ;11763 =00
                                        ;11764          ;00-------------------; RESERVED OPERAND
U 0588, 0C80,0036,4030,0047,00FF,8      ;11765          NEXT/IE.OPER.FAULT      ;
                                        ;11766
                                        ;11767          ;01-------------------; SRC < 0
                                        ;11768          WB_M[MDR].AND.ZLIT12[8], ; IS THE 'ROUND' BIT SET
                                        ;11769          SET SIGN(FLAG0),        ; SRC IS NEGATIVE
                                        ;11770          WX.NE.0?,               ;
U 0589, 0141,2D52,0A70,4047,0058,C      ;11771          NEXT/FP.CVTDF.100       ; CONTINUE PROCESSING
                                        ;11772
                                        ;11773          ;10-------------------; SRC = 0
                                        ;11774          R[DST.R]_0,             ; WRITE A ZERO
                                        ;11775          CCOP1,                  ; SET CONDITION CODES
                                        ;11776          WRITE NOTREG,SIZE[LONG], ;
U 058A, 0C84,05B7,012C,4DDA,003F,9      ;11777          IRD1                    ; END OF INSTRUCTION
                                        ;11778
                                        ;11779          ;11-------------------; SRC > 0
                                        ;11780          WB_M[MDR].AND.ZLIT12[8], ; IS THE 'ROUND' BIT SET
                                        ;11781          WX.NE.0?,               ;
U 058B, 0181,2D52,0A70,4047,0058,C      ;11782          NEXT/FP.CVTDF.100       ; CONTINUE PROCESSING
                                        ;11783
                                        ;11784 =00
                                        ;11785 FP.CVTDF.100:
                                        ;11786          ;00-------------------;
                                        ;11787          R[DST.R]_M[TEMP1],      ; BIT NOT SET NO ROUNDING
                                        ;11788          CCOP1,                  ; SET CONDITION CODES
                                        ;11789          WRITE NOTREG,SIZE[LONG], ;
U 058C, 0884,1592,412C,4DDA,003F,9      ;11790          IRD1                    ; END OF INSTRUCTION
                                        ;11791
                                        ;11792          ;01-------------------;
                                        ;11793          M[TEMP1]_UNPACK(MB R[ZERO]), ; GET FRACTION DONT NEED SECOND HALF
                                        ;11794          SIGN(FLAG0)?,PUSH,      ; SRC NEGATIVE SET D TO SIGN
U 058D, 0886,1677,043D,8047,046D,8      ;11795          NEXT/FP.DSIGNSET        ;
                                        ;11796
                                        ;11797          ;10-------------------;
                                        ;11798          M[TEMP1]_MB+ZLIT0[80],  ; ROUND THE FRACTION
U 058E, 0986,1C11,06F4,0047,0851,5 377* ;11799          WB<31-30>?              ; WAS THERE CARRY INTO OVERFLOW BIT
                                        ;11800 =
```

```
                                ;11801 =01
                                ;11802          ;01-----------------------------; BIT 31 NOT SET NO CARRY
                                ;11803          R[TEMP1]_RB.ASL.2,              ; SL 2 TO REMOVE HIDDEN & OVERFLOW BIT
U 0515, 0884,05F7,0020,4047,0054,F ;11804       NEXT/FP.CVTPACK                 ; PACK AND WRITE
                                ;11805 =11
                                ;11806          ;11-----------------------------; BIT 31 SET INCREMENT EXPONENT
                                ;11807          R[TEMP5].SIZ_RB+1,              ; BUMP EXPONENT
U 0517, 0082,0E7C,0011,4047,0054,8 ;11808       SIZE[WORD]                      ;
                                ;11809
                                ;11810          ;-------------------------------;
                                ;11811          WB_M[TEMP5].AND.ZLIT8[OFF],     ; CHECK FOR OVERFLOW
                                ;11812          SIZE[WORD],                     ; AND WITH FF00
U 0548, 0D80,5D92,0A57,F847,0068,6 ;11813       WX.NE.0?                        ;
                                ;11814 =0
                                ;11815          ;0-----------------------------; NO OVERFLOW
                                ;11816          R[TEMP1]_RB.ASL.1,              ; SL1  REMOVE HIDDEN BIT
U 0686, 0884,05F7,0010,4047,0054,F ;11817       NEXT/FP.CVTPACK                 ; PACK THEN WRITE
                                ;11818
                                ;11819          ;1-----------------------------; OVERFLOW
U 0687, 0C80,0036,4030,0047,00FF,B ;11820       NEXT/IE.FLOV.FAULT              ; TAKE THE FLOATING FAULT PATH
                                ;11821
                                ;11822 FP.CVTPACK:
                                ;11823          ;-------------------------------;
                                ;11824          M[TEMP1]_D.OR.PACK(MB R[TEMP5]),; PACK UP
U 054F, 0886,16B2,4031,4047,0058,C ;11825       NEXT/FP.CVTDF.100               ;
```

```
;11826 .TOC "  Floating point and CRC          : GENERAL PACK AND WRITE ROUTINES"
;11827
;11828 ;*****************************************************************************
;11829 ;
;11830 ;                          PACK AND  WRITE  SUBROUTINES
;11831 ;
;11832 ;       FP.PACK.T2T5
;11833 ;
;11834 ;               Input           Temp2           fraction (bits 0-31)
;11835 ;                               Temp5           exponent
;11836 ;
;11837 ;               Resources:
;11838 ;                               D               if answer is negative
;11839 ;
;11840 ;               Operation:
;11841 ;                       Pack up bits 0-31 of fraction with exponent
;11842 ;                       OR'ing with D negative sign postion if negative
;11843 ;                       Entered by branching on the sign flag
;11844 ;                       exit to FP.WRITE.FIRST
;11845 ;
;11846 ;       FP.WRITE23
;11847 ;
;11848 ;               Input           Temp2 (bits 0-31)       floating point value
;11849 ;
;11850 ;               Resources
;11851 ;                               CCOP1                   to set condition
;11852 ;                                                       codes
;11853 ;               Operation:
;11854 ;                       If 2 operand instruction write operand (size
;11855 ;                       dependent) and exit
;11856 ;                       If 3 operand get dst via OS.WRT1 and return to
;11857 ;                       write out operand as above
;11858 ;
;11859 ;
;11860 ;       FP.WRITE.FIRST
;11861 ;       FP.WRITE.SEC
;11862 ;               Input           Temp2                   fraction for single
;11863 ;                               Temp2                   fraction for double
;11864 ;                               MDR/Temp3               bits 32-63 of fraction
;11865 ;                               Flag1                   write out flag
;11866 ;                                  (WRITE(FLAG1))
;11867 ;
;11868 ;               Operation:
;11869 ;                       For single and double operands
;11870 ;                       if single write and exit
;11871 ;                       if double write from Temp2 (CCOP1) and bump VA
;11872 ;                       test if writing from Temp3 or MDR
;11873 ;                       and act accordingly
;11874 ;
;11875 ;*****************************************************************************
```

```
                                ;11876 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;11877 =0
                                ;11878 FP.PACK.T2T5:
                                ;11879          ;0------------------------------; ANS IS POSITIVE BIT 15=0
                                ;11880          M[TEMP2]_PACK(MB R[TEMP5]),     ;
                                ;11881          IR<5>?,                         ; FLOAT OR DOUBLE WRITE
U 0688, 0886,26B7,08F1,4047,0050,8 ;11882       NEXT/FP.WRITE.FIRST             ;
                                ;11883
                                ;11884          ;1------------------------------; ANS IS NEG
U 0689, 0D80,0D77,2030,4047,0055,0 ;11885       D_ZLIT12[8]                     ; SET UP TO SET SIGN BIT
                                ;11886
                                ;11887          ;--------------------------------;
                                ;11888          M[TEMP2]_D.OR.PACK(MB R[TEMP5]),;
                                ;11889          IR<5>?,                         ; FLOAT OR DOUBLE WRITE
U 0550, 0C86,26B2,48F1,4047,0050,8 ;11890       NEXT/FP.WRITE.FIRST             ;
                                ;11891
                                ;11892 =0000
                                ;11893 =0110
                                ;11894 FP.WRITE23:
                                ;11895          ;0110------------------------------; 2 OPERAND SINGLE WRITE
                                ;11896          R[DST.R]_M[TEMP2],              ; WRITE
                                ;11897          WRITE NOTREG,SIZE[IDEP],        ; SIZE DEPENDS ON INSTRUCTION
                                ;11898          CCOP1,                          ; SET CONDITION CODES
U 0506, 0C84,2592,413C,4DDA,003F,9 ;11899       IRD1                            ; DONE
                                ;11900
                                ;11901          ;0111------------------------------; 3 OPERAND GET THIRD AND RET
                                ;11902          LOD INC BRA?,                   ;
U 0507, 0C80,0036,41B0,0047,0414,0 ;11903       PUSH,NEXT/OS.WRT1               ;
                                ;11904
                                ;11905 FP.WRITE.FIRST:
                                ;11906          ;1000------------------------------; SINGLE WRITE
                                ;11907          R[DST.R]_M[TEMP2],              ; WRITE
                                ;11908          WRITE NOTREG,SIZE[IDEP],        ; SIZE DEPENDS ON INSTRUCTION
                                ;11909          CCOP1,                          ; SET CONDITION CODES
U 0508, 0C84,2592,413C,4DDA,003F,9 ;11910       IRD1                            ; DONE
                                ;11911
                                ;11912          ;1001------------------------------; DOUBLE WRITE
                                ;11913          R[DST.R]_M[TEMP2],              ; WRITE
                                ;11914          WRITE NOTREG,SIZE[IDEP],        ;
                                ;11915          CCOP1,                          ; SET CONDITION CODES
                                ;11916          SET MM.NOINT,                   ; NO INTERRUPTS BETWEEN WRITES
U 0509, 0064,2592,403C,4DDA,0056,F ;11917       NEXT/FP.WRITE.SEC               ; FINISH UP WRITE
                                ;11918 =
```

```
                                ;11919 FX.WRITE.SEC:                            ; ENTRY G H FLOAT
                                ;11920 FP.WRITE.SEC:                            ;
                                ;11921          ;-------------------------------; SECOND PART OF DOUBLE WRITE
                                ;11922          VA_VA+4,                        ; BUMP THE VA
U 056F, 0C80,0036,45F0,0447,0053,0  ;11923      WRITE(FLAG1)?                   ; SECOND WRITE FROM MDR OR TEMP3
                                ;11924
                                ;11925 =0*
                                ;11926 FX.WRITE.SEC.10:                         ; SECONDARY ENTRY G H FLOAT
                                ;11927          ;0*-------------------------------; TEMP3 WRITE
                                ;11928          R[DST.R+1]_M[TEMP3],            ; WRITE SECOND HALF
                                ;11929          WRITE NOTREG, SIZE[LONG],       ;
U 0530, 0884,3592,412F,45DA,003F,9  ;11930      IRD1                            ;
                                ;11931
                                ;11932          ;1*-------------------------------; MDR WRITE
                                ;11933          R[DST.R+1]_M[MDR],              ;
                                ;11934          WRITE NOTREG,SIZE[LONG],        ;
U 0532, 0885,2592,412F,45DA,003F,9  ;11935      IRD1                            ;
```

```
                                        ;11936 .TOC "  Floating point and CRC          : ADDF2 ADDF3"
                                        ;11937
                                        ;11938 ;*****************************************************************************
                                        ;11939 ;
                                        ;11940 ;       40      ADDF2   add.rf,sum.mf
                                        ;11941 ;
                                        ;11942 ;       41      ADDF3   add1.rf,add2.rf,sum.wf
                                        ;11943 ;
                                        ;11944 ;               Input           Q               src1 (add1.rf)
                                        ;11945 ;                               MDR             src2 (add2.rf/sum.mf)
                                        ;11946 ;               Resources:
                                        ;11947 ;                       for src1                Q packed
                                        ;11948 ;                               Temp2           fraction
                                        ;11949 ;                               Temp5           exponent
                                        ;11950 ;                               Flag0           sign of operand
                                        ;11951 ;                                  (ADD1(FLAG0))
                                        ;11952 ;
                                        ;11953 ;                       for src2                MDR packed
                                        ;11954 ;                               Temp0           fraction
                                        ;11955 ;                               Temp7           exponent
                                        ;11956 ;                               Flag1           sign of operand
                                        ;11957 ;                                  (ADD2(FLAG1))
                                        ;11958 ;               Subroutines:
                                        ;11959 ;                               FP.ADDFOP1      parse src1
                                        ;11960 ;                               FP.ADDFOP2      parse src2
                                        ;11961 ;
                                        ;11962 ;               Exit:
                                        ;11963 ;                               FP.WRITE23
                                        ;11964 ;               Operation:
                                        ;11965 ;                       This is the driving routine for the single
                                        ;11966 ;                       precision ADDF2 and ADDF3 instructions
                                        ;11967 ;
                                        ;11968 ;       **********FPA ENTRY AND INTERFACE**********
                                        ;11969 ;               (OS.RED) - FI.ADDF2.REG/FI.ADDF2.MEM
                                        ;11970 ;               (OS.RED) - FI.ADDF3.REG/FI.ADDF3.MEM
                                        ;11971 ;
                                        ;11972 ;*****************************************************************************
                                        ;11973 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;11974 =00
                                        ;11975 FP.ADDF23:                               ; **********************
                                        ;11976         ;00-------------------------------;
                                        ;11977         M[TEMP2]_Q,                     ; MOVE SRC1 TO AN MBUS SOURCE
U 0088, 0086,203A,403D,8047,045A,1      ;11978         PUSH,NEXT/FP.ADDFOP1            ; GO PARSE SRC1
                                        ;11979
                                        ;11980 FP.ADDF.2:
                                        ;11981         ;01-------------------------------;
                                        ;11982         R[TEMP7]_EXP(M[MDR]),           ; START PARSE OF SRC2
                                        ;11983         FRO.FLTZ?,                      ; TEST VALIDTY
U 0089, 0C85,2437,0AB1,C047,0C59,8 323* ;11984         PUSH,NEXT/FP.ADDFOP2            ; RETURNS AT END OF OPERATION
                                        ;11985
                                        ;11986         ;10-------------------------------;
                                        ;11987         IR<2-0>?,                       ; 2 OR 3 OPERAND INSTRUCTION
U 008A, 0080,0036,4670,0047,0050,6      ;11988         NEXT/FP.WRITE23                 ; DO WRITE OR GET DST ADDRESS
                                        ;11989 =
```

```
                        ;11990 .TOC "  Floating point and CRC          : SUBF2 SUBF3"
                        ;11991
                        ;11992 ;*****************************************************************************
                        ;11993 ;
                        ;11994 ;       42      SUBF2   sub.rf,dif.mf
                        ;11995 ;
                        ;11996 ;       43      SUBF3   sub.rf,min.rf,dif.wf
                        ;11997 ;
                        ;11998 ;               Input
                        ;11999 ;                       MDR             sub.rf
                        ;12000 ;               Resources:
                        ;12001 ;                       for src1 (sub.rf)
                        ;12002 ;                               Temp2           fraction
                        ;12003 ;                               Temp5           exponent
                        ;12004 ;                               Flag0           sign of operand
                        ;12005 ;                                  (ADD1(FLAG0))
                        ;12006 ;                       for src2 (dif.mf,min.rf)
                        ;12007 ;                               Temp0           fraction
                        ;12008 ;                               Temp7           exponent
                        ;12009 ;                               Flag1           sign of operand
                        ;12010 ;                                  (ADD2(FLAG1))
                        ;12011 ;               Subroutines:
                        ;12012 ;                               OS.MOD          to get src2 for SUBF2
                        ;12013 ;                               OS.FRED         to get src2 for SUBF3
                        ;12014 ;               Exit:
                        ;12015 ;                               FP.ADDF.2       to join common flows with
                        ;12016 ;                                               ADDF2,ADDF3
                        ;12017 ;               Operation:
                        ;12018 ;                       Entry after have src1
                        ;12019 ;                       flip the sign bit and get the second
                        ;12020 ;                       operand returning through IRDROM
                        ;12021 ;                       Validate the first operand and parse it
                        ;12022 ;                       and then join common flows with
                        ;12023 ;                       ADDF2,ADDF3 at FP.ADDF.2
                        ;12024 ;
                        ;12025 ;       **********FPA ENTRY AND INTERFACE**********
                        ;12026 ;       SUBF2   (OS.RLD) - FI.ADDF2.REG/FI.ADDF2.MEM
                        ;12027 ;       SUBF3   (OS.RED) - FI.ADDF3.REG/FI.ADDF3.MEM
                        ;12028 ;
                        ;12029 ;*****************************************************************************
                        ;12030 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;12031 FP.SUBF2:                               ; **********
                        ;12032         ;-------------------------------;
                        ;12033         MDR_M[MDR].XOR.ZLIT12[8],       ; FLIP THE SIGN BIT
                        ;12034         LOD_INC BRA?,                   ; NEED THIS APPROACH
U 0345, 0D81,2D53,41B0,4467,0011,0      ;12035         NEXT/OS.MOD                     ; GET SECOND OPERAND
                        ;12036                                         ; RET THRU IRDROM TO FP.SUBF23.20
                        ;12037
                        ;12038 FP.SUBF3:                               ; **********
                        ;12039         ;-------------------------------;
                        ;12040         MDR_M[MDR].XOR.ZLIT12[8],       ; FLIP THE SIGN BIT
                        ;12041         LOD_INC BRA?,                   ; NEED THIS APPROACH
U 0349, 0581,2D53,41B0,4467,0016,0      ;12042         NEXT/OS.FRED                    ; GET SECOND OPERAND
                        ;12043                                         ; RET THRU IRDROM TO FP.SUBF23.20
```

```
                                ;12044 FP.SUBF23.20:
                                ;12045          ;--------------------------------;
U 0351, 0886,203A,403D,8047,0058,F ;12046       M[TEMP2]_Q                      ; MOVE SRC1 TO THE WORKING TEMP
                                ;12047
                                ;12048 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;12049 FP.SUBF.100:
                                ;12050          ;--------------------------------;
                                ;12051          R[TEMP5]_EXP(M[TEMP2]),         ; SIGN HAS BEEN FLIPPED
U 058F, 0C84,2437,0AB1,4047,0859,0 323* ;12052  FRO.FLTZ?                        ; SPECIAL GROUP OF VALIDITY TEST
                                ;12053
                                ;12054 =00
                                ;12055          ;00--------------------------------; SRC1 = 0 SIGN FLIPPED
                                ;12056          R[TEMP2]_Q_0,                   ; MAKE IT CLEAN ZERO
                                ;12057          SET UPZERO?(FLAG3),             ; SET FLAG TO INDICATE ZERO
U 0590, 0C5C,05B7,1030,8047,0008,9 ;12058       NEXT/FP.ADDF.2                  ; FOR SRC1 CONTINUE PROCESSING
                                ;12059
                                ;12060          ;01--------------------------------; SRC1 < 0 SIGN FLIPPED
                                ;12061          M[TEMP2]_UNPACK(MB R[ZERO]),    ; UNPACK THE FRACTION
                                ;12062          SET ADD1?(FLAG0),               ; SET SIGN FLAG FOR SRC1
U 0591, 0C46,2677,003D,8047,0008,9 ;12063       NEXT/FP.ADDF.2                  ;
                                ;12064
                                ;12065          ;10--------------------------------; RESERVED OPERAND SIGN FLIPPED
U 0592, 0C80,0036,4030,0047,00FF,8 ;12066       NEXT/IE.OPER.FAULT              ;
                                ;12067
                                ;12068          ;11--------------------------------; SRC1 > 0 SIGN FLIPPED
                                ;12069          M[TEMP2]_UNPACK(MB R[ZERO]),    ; UNPACK THE FRACTION
U 0593, 0C86,2677,003D,8047,0008,9 ;12070       NEXT/FP.ADDF.2                  ;
```

```
;12071 .TOC "  Floating point and CRC          : FP.ADDFOP1 FP.ADDFOP2 "
;12072
;12073 ;*****************************************************************************
;12074 ;
;12075 ;       SINGLE PRECISION OPERAND PARSERS
;12076 ;
;12077 ;               FP.ADDFOP1
;12078 ;
;12079 ;                       Input
;12080 ;                               Temp2           packed operand
;12081 ;                               Flag0   clear
;12082 ;                                  (ADD1(FLAG0))
;12083 ;                               Flag1           clear
;12084 ;                                  (ADD2(FLAG1))
;12085 ;
;12086 ;                       Resources:
;12087 ;                               Temp2           fraction unpacked
;12088 ;                                                 with overflow and hidden bit
;12089 ;                                                 in bits 31-30
;12090 ;                               Temp5           exponent (biased)
;12091 ;                               Flag0   sign of operand
;12092 ;                                  (ADD1(FLAG0))
;12093 ;                               Flag3           set if operand zero
;12094 ;                                  (OPZERO(FLAG3))
;12095 ;                               Q               zero if operand is zero
;12096 ;
;12097 ;
;12098 ;               FP.ADDFOP2
;12099 ;
;12100 ;                       Input
;12101 ;                               MDR             packed operand
;12102 ;                               Flag3           state of src1
;12103 ;                                  (OPZERO(FLAG3))
;12104 ;                               Flag1           clear
;12105 ;                                  (ADD2(FLAG1))
;12106 ;                               Q               packed src1 operand
;12107 ;
;12108 ;                       Resources:
;12109 ;                               Temp0           fraction
;12110 ;                               Flag1           sign of operand
;12111 ;                                  (ADD2(FLAG1))
;12112 ;
;12113 ;
;12114 ;                       Operation:
;12115 ;                               Operand is parsed if not zero
;12116 ;                               and not reserved then exit is to
;12117 ;                               FP.ADDf.SUB to continue the add function
;12118 ;                               If OPZERO(FLAC3) indicated that
;12119 ;                               src1 was zero then the answer is src2 and
;12120 ;                               is found packed in MDR
;12121 ;
;12122 ;
;12123 ;*****************************************************************************
```

```
                                ;12124 FX.ADDFOP1:                              ; ENTRY FOR GH FLOAT
                                ;12125 FP.ADDFOP1:                              ;
                                ;12126          ;-------------------------------;
                                ;12127          R[TEMP5] EXP(M[TEMP2]),         ; GET EXPONENT TEST VALIDITY
U 05A1, 0484,2437,0AB1,4047,0859,4 323*  ;12128          FRO.FLTZ?                       ;
                                ;12129
                                ;12130 =00
                                ;12131          ;00-----------------------------; RESERVED OPERAND
U 0594, 0C80,0036,4030,0047,00FF,8      ;12132          NEXT/IE.OPER.FAULT              ;
                                ;12133
                                ;12134          ;01-----------------------------; SRC1 < 0
                                ;12135          M[TEMP2] UNPACK(MB R[ZERO]),    ; UNPACK THE FRACTION
                                ;12136          SET ADD1?(FLAG0),               ; SET SIGN FLAG FOR FIRST OPERAND
U 0595, 0846,2677,00BD,8047,0000,1      ;12137          RETURN [+1]                     ;
                                ;12138
                                ;12139          ;10-----------------------------; SRC1 = 0
                                ;12140          R[TEMP2] Q_0,                   ; BOTH PACKED & UNPACKED CLEAN ZERO
                                ;12141          SET OPZERO?(FLAG3),             ; SET FLAG INDICATING SRC1=0
U 0596, 085C,05B7,10B0,8047,0000,1      ;12142          RETURN [+1]                     ; GO BACK FOR SRC2
                                ;12143
                                ;12144          ;11-----------------------------; SRC1 > 0
                                ;12145          M[TEMP2] UNPACK(MB R[ZERO]),    ; UNPACK SRC1
U 0597, 0886,2677,00BD,8047,0000,1      ;12146          RETURN [+1]                     ;
                                ;12147
                                ;12148 =00
                                ;12149 FP.ADDFOP2:
                                ;12150          ;00-----------------------------; RESERVED OPERAND
U 0598, 0C80,0036,4030,0047,00FF,8      ;12151          NEXT/IE.OPER.FAULT              ;
                                ;12152
                                ;12153          ;01-----------------------------; SRC2 < 0
                                ;12154          MTEMP0 UNPACK(MDR R[ZERO]),     ; UNPACK FRACTION SRC2
                                ;12155          SET ADD2(FLAG1),                ; SET SIGN FLAG FOR SRC2
                                ;12156          OPZERO(FLAG3)?,                 ; WAS SRC1 ZERO
U 0599, 0C4F,2677,04FD,8047,0059,C      ;12157          NEXT/FP.ADDF.SUB                ; CONTINUE PROCESSING MAYBE
                                ;12158
                                ;12159          ;10-----------------------------; SRC2 = 0
                                ;12160          M[TEMP2] Q,                     ; ANSWER IS SRC1 PACKED IN Q
                                ;12161          CLEAR OPZERO(FLAG3),            ; CLEAR SO NOT TAKEN AS OVERFLOW
U 059A, 041E,203A,40BD,8047,0000,1      ;12162          RETURN [+1]                     ; ALL DONE
                                ;12163
                                ;12164          ;11-----------------------------; SRC2 > 0
                                ;12165          MTEMP0 UNPACK(MDR R[ZERO]),     ; UNPACK SRC2 FRACTION
                                ;12166          OPZERO?(FLAG3)?,                ; WAS SRC1 ZERO
U 059B, 0487,2677,04FD,8047,0059,C      ;12167          NEXT/FP.ADDF.SUB                ; CONTINUE PROCESSING
```

```
;12168 .TOC "  Floating point and CRC          : FP.ADDF.SUB "
;12169
;12170 ;*****************************************************************************
;12171 ;
;12172 ;       FP.ADDF.SUB
;12173 ;
;12174 ;               INTERFACE FOR ADDF2/3,SUBF2/3,ACBF FOR ADDRTN ROUTINE
;12175 ;
;12176 ;
;12177 ;               Input
;12178 ;                                       Temp5           src1 biased exponert
;12179 ;                                       Temp7           src2 biased exponent
;12180 ;                                       MDR             packed  src2
;12181 ;                                       Q               packed src1
;12182 ;                                   *   Temp2           src1 fraction
;12183 ;                                   *   Temp0           src2 fraction
;12184 ;
;12185 ;                                       Flag3           set if src1 zero
;12186 ;                                          (OPZERO(FLAG3))
;12187 ;
;12188 ;                                   * form of fraction
;12189 ;                                       bit 31 overflow bit
;12190 ;                                       bit 30 hidden bit
;12191 ;                                       binary point before bit 30
;12192 ;
;12193 ;               Resources:
;12194 ;                                       Platch          difference between exponents
;12195 ;                                       D               difference between exponents
;12196 ;                                       Temp2           if difference >31
;12197 ;                                                       contains packed answe:
;12198 ;                                                       otherwise  normalized fraction
;12199 ;                                       Temp5           exponent of answer
;12200 ;                                       Flag3           cleared
;12201 ;                                          (OP.ZERO(FLAG3))
;12202 ;
;12203 ;               Subroutines:
;12204 ;                                       FP.ADDRTN       to perform the add function
;12205 ;
;12206 ;
;12207 ;
;12208 ;               Exit:
;12209 ;                                       FP.NORM.SNG     to normalize round and
;12210 ;                                                       pack the answer
;12211 ;                                                       no return if floating overflow
;12212 ;                                                       or FU bit set and underflow
;12213 ;                                       **return from this routine is to the
;12214 ;                                         main calling routine
;12215 ;
;12216 ;
;12217 ;*****************************************************************************
```

```
                                        ;12218 =00
                                        ;12219 FP.ADDF.SUB:
                                        ;12220          ;00-----------------------------; OPZERO(FLAG3) NOT SET  SRC1 NOT ZERO
                                        ;12221          PL_D_M[TEMP5]-R[TEMP7],         ; COMPUTE DIFFERENCE IN EXPONENTS
                                        ;12222          SIGND CMP?,SIZE[WORD],          ; DETERMINE WHICH IS GREATER
                                        ;12223          PUSH,                           ; SAVE RETURN
U 059C, 0880,5BC0,2B51,C047,0C51,0 374* ;12224          NEXT/FP.ADDRTN                  ; OFF TO HEART OF ADD ROUTINE
                                        ;12225
                                        ;12226          ;01-----------------------------; OPZERO(FLAG3) SET SRC1 WAS ZERO
                                        ;12227          R[TEMP2]_M[MDR],                ; ANSWER IS SRC2
                                        ;12228          CLEAR OPZERO(FLAG3),            ;
U 059D, 041D,2592,40B0,8047,0000,1      ;12229          RETURN [+1]                     ; ALSO RETURN +1 FROM FP.ADDRTN
                                        ;12230
                                        ;12231          ;10-----------------------------; RETURN +2 FROM FP.ADDRTN
                                        ;12232          M[TEMP2]_Q,                     ; ANSWER IS SRC1
U 059E, 0C86,203A,40BD,8047,0000,1      ;12233          RETURN [+1]                     ; BACK TO CALLING ROUTINE
                                        ;12234
                                        ;12235          ;11-----------------------------; RETURN +3 FROM FP.ADDRTN
                                        ;12236          M[TEMP5]_MB-ZLIT0[1E],          ; SUB FACTOR TO EXPONENT
                                        ;12237          SIZE[WORD],                     ; CANT OVERFLOW YET
U 059F, 0D86,5C10,0010,F047,0068,C      ;12238          NEXT/FP.NORM.SNG                ; OFF TO NORMALIZATION ROUTINE
                                        ;12239                                          ; ANSWER IS UNPACKED IN
                                        ;12240                                          ; FRACTION IN TEMP2
                                        ;12241                                          ; EXPONENT IN TEMP5
                                        ;12242                                          ; SIGN SETTING D=0/8000
                                        ;12243                                          ; FINAL RETURN TO CALLING INSTRUCTION
                                        ;12244                                          ; EXCEPT IF FLOATING OVERFLOW OR
                                        ;12245                                          ; FU BIT SET AND FLOATING UNDERFLOW
                                        ;12246                                          ; INSTRUCTION THEN FAULTS
```

```
;12247 .TOC "  Floating point and CRC          : FP.ADDRTN "
;12248
;12249 ;*****************************************************************************
;12250 ;
;12251 ;       FP.ADDRTN
;12252 ;               HEART OF THE ADD SINGLE PRECISION ROUTINE
;12253 ;                       USED BY ADDF2,ADDF3,SUBF2,SUBF3,ACBF,POLYF
;12254 ;
;12255 ;
;12256 ;
;12257 ;               Input:
;12258 ;                                       D               exponent difference
;12259 ;                                       Platch          exponent difference
;12260 ;                                       Temp2           src1 fraction
;12261 ;                                       Temp5           src1 exponent
;12262 ;                                       Flag0           sign of src1
;12263 ;                                          (ADD1(FLAG0))
;12264 ;                                       Temp0           src2 fraction
;12265 ;                                       Temp7           src2 fraction
;12266 ;                                       Flag1           sign of src2
;12267 ;                                          (ADD2(FLAG1))
;12268 ;
;12269 ;               Resources:
;12270 ;                                       Temp1           working temp
;12271 ;                                       Flag5           diff sign flag
;12272 ;                                          (SAMESIGN(FLAG4))
;12273 ;
;12274 ;               Subroutines:
;12275 ;                                       FP.ADDSIGNOP1
;12276 ;                                       FP.ADDSIGNOP2
;12277 ;
;12278 ;               Output
;12279 ;                                       Temp2           fraction of answer
;12280 ;                                       Temp5           exponent of answer
;12281 ;                                       D               sign of answer
;12282 ;
;12283 ;               Operation:
;12284 ;                       Enters on a 3 way branch testing exponent relationship
;12285 ;                                               src1 > src2
;12286 ;                                               src1 = src2
;12287 ;                                               src1 < src2
;12288 ;
;12289 ;                       In the case of unequal exponents
;12290 ;                       If the difference is > 31 then a return is effected
;12291 ;                       (+1 for src2 is answer and +2 if src1 is answer)
;12292 ;
;12293 ;                       Note all other returns are +3 and assume the answer is
;12294 ;                       the unpacked fraction in Temp2 , exponent in Temp5
;12295 ;                       and sign positioned in D
;12296 ;                       If the difference is < 31 then the fractional
;12297 ;                       part of the smaller value is placed in Temp1 and the
;12298 ;                       appropriate sign routine is called (FP.ADDSIGNOP1,
;12299 ;                       FP.ADDSIGNOP2) to determine sign of answer
;12300 ;                       and to complement the smaller value if appropriate
```

```
                                ;12301 ;                        In the case of similar exponents src2 is selected as
                                ;12302 ;                        the "smaller" value the routine FP.ADDSIGNOP1
                                ;12303 ;                        is called
                                ;12304 ;                        note only with like signs can the sign of the
                                ;12305 ;                        answer be predetermined
                                ;12306 ;                        therefore, a flag (SAMESIGN(FLAG4)) is used if the
                                ;12307 ;                        signs are unlike and exponents are same
                                ;12308 ;
                                ;12309 ;                        Upon return
                                ;12310 ;                                with unlike exponents
                                ;12311 ;                        the smaller value is aligned as appropriate
                                ;12312 ;                        and the fractions are added and the
                                ;12313 ;                        larger exponent value is selected as the
                                ;12314 ;                        correct one and an exit is effected
                                ;12315 ;
                                ;12316 ;                                with like exponents
                                ;12317 ;                        Naturally no alignment occurs
                                ;12318 ;                        therefore if the signs are the same the
                                ;12319 ;                        fractions are added and a return is made
                                ;12320 ;                        if the signs are unlike the fractions are added
                                ;12321 ;                        if bit 31 is set then the value is complemented
                                ;12322 ;                        and the sign flag in D inverted exit is then made
                                ;12323 ;                        this is possible mathematically since no alignment
                                ;12324 ;                        has occurred
                                ;12325 ;
                                ;12326 ;*************************************************************************
                                ;12327
                                ;12328 =000
                                ;12329 FP.ADDRTN:
                                ;12330          ;000-----------------------------; SRC1 > SRC2 (EXPONENTS)
                                ;12331          D_ZLIT0[31.]-D,                 ; IS EXP DIFFERENCE > 31
                                ;12332          WB<31-30>?,                     ;
U 0510, 0580,0C33,26F0,F847,0851,9 377*  ;12333          NEXT/FP.ADDRTN.100              ;
                                ;12334
                                ;12335          ;001-----------------------------; SRC1 = SRC2
                                ;12336          R[TEMP1]_M[TEMP0],              ; EXPS ARE EQUAL
                                ;12337          ADD2(FLAG1) ADD1(FLAG0)?,       ; TEST FOR SIGN OF ANSWER
U 0511, 0084,0592,4530,4047,045B,0       ;12338          PUSH,NEXT/FP.ADD.SIGNOP1        ; GET SIGN OF ANS IN D
                                ;12339
                                ;12340          ;010-----------------------------; SRC1 < SRC2 (EXPONENTS)
                                ;12341          D_D+ZLIT0[31.],                 ; IS DIFFERENCE IN EXPONENTS >31
                                ;12342          WB<31-30>?,                     ;
U 0512, 0180,0C31,26F0,F847,0854,5 377*  ;12343          NEXT/FP.ADDRTN.200              ;
                                ;12344
                                ;12345 =100
                                ;12346          ;100-----------------------------; RETURN +3 FROM FP.ADD.SIGN
                                ;12347          R[TEMP0]_M[TEMP1],              ;
U 0514, 0884,1592,4470,0047,0068,A       ;12348          SAMESIGN(FLAG4)?                ; IF SIGNS SAME NO CHECK
                                ;12349 =
```

```
                                ;12350 =0
                                ;12351          ;0------------------------------;
                                ;12352          R[TEMP2]_M[TEMP0]+RB,           ; THIS IS THE ADD
                                ;12353          WB<31-30>?,                     ; WAS THE SIGN BIT SET
U 068A, 0C84,0001,06F0,8047,0851,D 338*         ;12354          NEXT/FP.ADDRTN.30               ; FOR THE UNLIKE SIGNS OF EQU EXP
                                ;12355
                                ;12356 FP.ADDRTN.20:
                                ;12357          ;1------------------------------;
                                ;12358          R[TEMP2]_M[TEMP0]+RB,           ; THIS IS THE ADD
                                ;12359          CLEAR SAMESIGN(FLAG4),          ;
U 068B, 0024,0001,00B0,8047,0000,3              ;12360          RETURN [+3]                     ;
                                ;12361
                                ;12362 =01
                                ;12363 FP.ADDRTN.30:
                                ;12364          ;01-----------------------------;
U 051D, 0080,0036,40B0,0047,0000,3              ;12365          RETURN [+3]                     ; DONE DONT CHANGE
                                ;12366
                                ;12367          ;11-----------------------------;
U 051F, 0C86,2003,003D,8047,005A,2              ;12368          M[TEMP2]_-MB                    ; ONLY IF NO ALIGNMENT CAN WE GET HERE
                                ;12369
                                ;12370          ;------------------------------;
                                ;12371          D_D.XOR.ZLIT12[8],              ; FLIP THE SIGN BIT
U 05A2, 0980,0D73,60B0,4047,0000,3              ;12372          RETURN [+3]                     ;
                                ;12373
                                ;12374 =000
                                ;12375 =001
                                ;12376 FP.ADDRTN.100:
                                ;12377          ;001----------------------------; SRC1 > SRC2
                                ;12378          R[TEMP1]_M[TEMP0],              ; SMALLER FRACT (SRC2) INTO WORKING TEMP
                                ;12379          ADD2(FLAG1) ADD1(FLAG0)?,       ; CHECK SIGN COMPLEMENT IF NECESSARY
U 0519, 0084,0592,4530,4047,045B,0              ;12380          PUSH,NEXT/FP.ADD.SIGNOP1        ; DIFFERENCE < 31
                                ;12381
                                ;12382 =011
                                ;12383          ;011----------------------------; DIFFERENCE >31
                                ;12384          PL_[1F],                        ; SET TO 31
U 051B, 0980,0EF6,4030,F847,0051,9              ;12385          NEXT/FP.ADDRTN.100              ; NEED FOR ACCURACY POLYF
                                ;12386
                                ;12387 =100
                                ;12388          ;100----------------------------; RETURN +3 FROM FP.ADD.SIGN
                                ;12389          R[TEMP0]_M[TEMP1].ASR.P,        ; ALIGN SRC2
U 051C, 0484,10F7,0030,0047,0068,B              ;12390          NEXT/FP.ADDRTN.20               ; CONTINUE PROCESSING
                                ;12391 =
                                ;12392 =01
                                ;12393 FP.ADDRTN.200:
                                ;12394          ;01-----------------------------; SRC2 > SRC1
                                ;12395          R[TEMP5]_M[TEMP7],              ; DIFFERENCE IS < 31
U 0545, 0084,7592,4031,4047,005A,0              ;12396          NEXT/FP.ADDRTN.250              ; SAVE CORRECT EXPONENT
                                ;12397
                                ;12398          ;11-----------------------------; DIFFERENCE >31
                                ;12399          PL_[01],                        ; SET TO 1 FOR -P SHIFT (IS 31)
U 0547, 0580,0EF6,4030,0847,0054,5              ;12400          NEXT/FP.ADDRTN.200              ; NEED FOR POLYF ACCURACY
```

```
                                        ;12401 =00
                                        ;12402 FP.ADDRTN.250:
                                        ;12403          ;00-----------------------------;
                                        ;12404          R[TEMP1] M[TEMP2],              ;
                                        ;12405          ADD2(FLAG1) ADD1(FLAG0)?,       ; TEST FOR SIGN
U 05A0, 0C84,2592,4530,4047,045B,4      ;12406          PUSH,NEXT/FP.ADD.SIGNOP2        ;
                                        ;12407
                                        ;12408 =11
                                        ;12409          ;11-----------------------------; RET +3 FROM FP.ADD.SIGNOP2
                                        ;12410          R[TEMP2] M[TEMP1].ASR.-P,       ; ALIGN SRC1
U 05A3, 0484,1AB7,0030,8047,0068,B      ;12411          NEXT/FP.ADDRTN.20               ; CONTINUE PROCESSING
                                        ;12412 =
```

```
;12413 .TOC "  Floating point and CRC          : NORMALIZATION ROUTINES "
;12414
;12415 ;*****************************************************************************
;12416 ;
;12417 ;       NORMALIZATION ROUTINES
;12418 ;
;12419 ;               FP.NORM.SNG  - ADDF2/3,SUBF2/3,ACBF,EMODF,POLYF
;12420 ;               FP.NORM.DBL  - ADDD2/3,SUBD2/3,ACBD,EMODD,POLYD
;12421 ;               FP.NORM.MUDI - MULF2/3,DIVF2/3,MULD2/3,DIVD2/3
;12422 ;
;12423 ;               Input:
;12424 ;                                       Temp2                   fraction bits 0-31
;12425 ;                                       Temp3                   fraction bits 32-63
;12426 ;                                                               (if double precision)
;12427 ;                                       D                       0/8000 sign of answer
;12428 ;                                       Temp5                   exponent and bias
;12429 ;                                                               plus factor for platch
;12430 ;                                                               and binary point
;12431 ;
;12432 ;               Resources:
;12433 ;                                       Platch
;12434 ;                                       D
;12435 ;
;12436 ;               Output
;12437 ;                                       Temp2,Temp3             packed answer
;12438 ;
;12439 ;               Operation:
;12440 ;
;12441 ;                       FP.NORM.MUDI
;12442 ;                               Entry for multiplication division instructions
;12443 ;                               to set the sign of the product or quotient
;12444 ;                               then branching on single/double precision
;12445 ;                               type instruction to..
;12446 ;
;12447 ;                       FP.NORM.SNG
;12448 ;                       FP.NORM.DBL
;12449 ;                               These entries are constrained to allow
;12450 ;                               conditional entry based on single/double
;12451 ;                               precision instruction types or they
;12452 ;                               be called individually where the type is known
;12453 ;
;12454 ;
;12455 ;                       For single precision
;12456 ;                               The find first set function determines
;12457 ;                               the position of the binary point
;12458 ;                               If no 1 bit is found ( possible in the case
;12459 ;                               of a zero fraction from add or subs) then a
;12460 ;                               return is effected with a zero answer
;12461 ;                               Otherwise, the value is normalized by extracting
;12462 ;                               and removing the hidden bit
;12463 ;                               the fraction is then rounded (by adding 100-hex
;12464 ;                               since the hidden bit has been removed)
;12465 ;                               the carry if present is added to the exponent
```

```
;12466 ;                    At FP.NORM.SNG.20
;12467 ;                         Platch is added to the exponent (Temp5)
;12468 ;                         biased with a "fudge" factor to account for
;12469 ;                         the binary point. The addition is in a word to
;12470 ;                         allow for over and under flow testing
;12471 ;                         If <0  underflow has occurred
;12472 ;                         FU bit is tested if set a fault exit is taken
;12473 ;                         to IE.FLUN.FAULT else fraction & exp are 0
;12474 ;                         If > 0  a check is made  if the value is
;12475 ;                         correctly represented in a byte
;12476 ;                         if overflow has occurred a fault exit is taken
;12477 ;                         to IE.FLOV.FAULT
;12478 ;                         If neither has occurred then the
;12479 ;                         fraction is packed with the exponent & the sign
;12480 ;                         and a return is effected
;12481 ;
;12482 ;                    For double precision
;12483 ;                         The find first function  gets the binary point
;12484 ;                         If the first 32 bits contain a 1 bit
;12485 ;                         the fraction is normalized by extraction
;12486 ;                         and removing the hidden bit
;12487 ;                         the low order (bits 32-63) are also extracted
;12488 ;                         the value is rounded (adding 100-hex
;12489 ;                         to bits 32-63 ) and propagating
;12490 ;                         the carry through bits 0-31 and into exponent
;12491 ;                         The double precision pack begins
;12492 ;                         with the low 9 bits of bits 0-31 being saved
;12493 ;                         in bits 32-63 , low order 32-63 are rotated
;12494 ;                         common flows continue with FP.NORM.SNG.20
;12495 ;
;12496 ;                         If the first 32 bits do not contain
;12497 ;                         a 1 bit then the second 32 bits are checked
;12498 ;                         If there is no 1 bit then the answer is 0
;12499 ;                         and a return is effected Else fraction is
;12500 ;                         normalized by extraction & removing hidden bit
;12501 ;                         the low order 32 bits are 0
;12502 ;                         an additional "fudge" factor of 32 is
;12503 ;                         subtracted from the Temp5
;12504 ;                         common flows continue with FP.NORM.SNG.20
;12505 ;
;12506 ;*****************************************************************************
```

```
                                ;12507 =10
                                ;12508 FP.NORM.MUDI:
                                ;12509          ;10-------------------------------; FOR MULF AND MULD,DIVF,DIVD
                                ;12510          D_ZLIT0[0],                       ; SET SIGN = 0
                                ;12511          CLEAR MUL1(FLAG2),                ; CLEAR FLAG FOR LATER USE
                                ;12512          IR<5>?,                           ; SINGLE OR DOUBLE
U 050A, 0510,0C37,28F0,0047,0068,C ;12513       NEXT/FP.NORM.SNG                  ;
                                ;12514
                                ;12515          ;11-------------------------------;
                                ;12516          D_ZLIT12[8],                      ; SET TO NEGATIVE
                                ;12517          CLEAR MUL1(FLAG2),                ; CLEAR FOR LATER USE
                                ;12518          IR<5>?,                           ;
U 050B, 0110,0D77,28F0,4047,0068,C ;12519       NEXT/FP.NORM.SNG                  ;
                                ;12520
                                ;12521 =0
                                ;12522 FP.NORM.SNG:
                                ;12523          ;0-------------------------------;
                                ;12524          PL_MSS M[TEMP2],                  ; FIND THE FIRST  BIT SET
                                ;12525          CLEAR MUL2(FLAG3),                ; CLEAR FLAG FOR LATER USE
                                ;12526          WX.NE.0?,                         ; WEED OUT ZERO FRACTION
U 068C, 0818,29C2,4A7D,8047,0068,E ;12527       NEXT/FP.NORM.SNG.10               ;
                                ;12528
                                ;12529 FP.NORM.DBL:
                                ;12530          ;1-------------------------------;
                                ;12531          PL_MSS M[TEMP2],                  ; FIND FIRST BIT SET FOR DOUBLE
                                ;12532          CLEAR MUL2(FLAG3),                ; CLEAR FLAG FOR LATER USE
                                ;12533          WX.NE.0?,                         ; WEED OUT ZERO FRACTION
U 068D, 0018,29C2,4A7D,8047,005A,8 ;12534       NEXT/FP.NORM.DBL.10               ;
                                ;12535
                                ;12536 =0
                                ;12537 FX.ZEROTEMP2:
                                ;12538 FP.ZEROTEMP2:
                                ;12539 FP.NORM.SNG.10:
                                ;12540          ;0-------------------------------; HANDLES 0 FRACTION FOR ADDS & SUBS
                                ;12541          R[TEMP2]_0,                       ; BE SURE ANS IS ZEROED
U 068E, 0C84,05B7,00B0,8047,0000,1 ;12542       RETURN [+1]                       ; ANSWER IS ZERO RETURN
                                ;12543
                                ;12544          ;1-------------------------------; NORMALIZE BY EXTRACTION
U 068F, 0486,2137,003D,8047,0069,0 ;12545       M[TEMP2]_(MB R[ZERO]).RR.P        ; STRIPS OFF HIDDEN BIT
                                ;12546
                                ;12547 =0
                                ;12548          ;0-------------------------------;
                                ;12549          M[TEMP2]_MB+ZLIT0[100],           ; ROUND THE FRACTION HIDDEN ALREADY GONE
U 0690, 0986,2C11,0038,0047,045C,D ;12550       PUSH,NEXT/FP.NORM.ADD.ALKC        ; GO GET CARRY
                                ;12551
                                ;12552 FP.NORM.SNG.20:
                                ;12553          ;1-------------------------------;
                                ;12554          M[TEMP5]_MB+PL,                   ; ADD PLATCH TO BIASED FACTORED EXPONENT
                                ;12555          SIZE[WORD],                       ; CHECKING FOR UNDER OR OVERFLOW
U 0691, 0086,5B11,0B50,0047,085A,4 410* ;12556  SIGND CMP?                         ; CHECK >,<,= 0
```

```
                                ;12557 =00
                                ;12558          ;00--------------------------------; > 0 MUST TEST FURTHER FOR OVERFLOW
                                ;12559          WB_M[TEMP5].AND.ZLIT8[0FF],     ; AND EXPONENT WITH FF00
                                ;12560          SIZE[WORD],                     ;
                                ;12561          WX.NE.0?,                       ; IF NON ZERO THEN OVERFLOW
U 05A4, 0D80,5D92,0A57,F847,0069,2 ;12562       NEXT/FP.DBL.PCK                 ;
                                ;12563
                                ;12564          ;01--------------------------------; UNDERFLOW CONDITION (=0)
                                ;12565          R[TEMP2]_0,                     ; SET RESULT TO 0 TENTATIVELY
U 05A5, 0884,05B7,0030,8047,005C,E ;12566       NEXT/FP.TESTPSL                 ; TEST IF FU SET
                                ;12567
                                ;12568          ;10--------------------------------; UNDERFLOW CONDITION (<0)
                                ;12569          R[TEMP2]_0,                     ; SET RESULT TO 0 TENTATIVELY
U 05A6, 0884,05B7,0030,8047,005C,E ;12570       NEXT/FP.TESTPSL                 ; TEST IF FU SET .
                                ;12571 =
                                ;12572 =0
                                ;12573 FP.DBL.PCK:
                                ;12574          ;0---------------------------------;
                                ;12575          M[TEMP2]_D.OR.PACK(MB R[TEMP5]),; PACK FRACTION OR WITH D SIGN BIT
U 0692, 0C86,26B2,40B1,4047,0000,1 ;12576       RETURN [+1]                     ;
                                ;12577
                                ;12578          ;1---------------------------------; OVERFLOW TIME
                                ;12579          VA_M[VA]-ZLIT0[4],              ; DECREMENT THE VA FOR POLYF,POLYD
U 0693, 0549,BC10,0030,24A7,00FF,B ;12580       SET FLAG1,NEXT/IE.FLOV.FAULT    ; FLAG1 FOR POLYD FPD RTN
                                ;12581
                                ;12582 =00
                                ;12583 FP.NORM.DBL.10:
                                ;12584          ;00--------------------------------; WX IS ZERO MAY BE ZERO ANS
                                ;12585          PL_MSS M[TEMP3],                ; FIND FIRST BIT SET IN SECOND HALF
                                ;12586          WX.NE.0?,                       ; IS THERE ONE
U 05A8, 0480,39C2,4A7D,8047,005A,C ;12587       NEXT/FP.NORM.DBL.50
                                ;12588
                                ;12589          ;01--------------------------------;
                                ;12590          M[TEMP2]_(MB R[TEMP3]).RR.P,    ; NORMALIZED FRACTION HIDDEN BIT GONE
U 05A9, 0C86,2137,0030,C047,046E,E ;12591       PUSH,NEXT/FP.T3ZERO             ;
                                ;12592
                                ;12593          ;10--------------------------------;
U 05AA, 0586,3C11,0038,0047,0069,4 ;12594       M[TEMP3]_MB+ZLIT0[100]          ; ROUND THE FRACTION
                                ;12595 =
                                ;12596 =0
                                ;12597          ;0---------------------------------;
                                ;12598          M[TEMP2]_MB+R[ZERO]+ALKC,       ; PROPAGATE THE CARRY
U 0694, 0486,2041,003D,8047,045C,D ;12599       PUSH,NEXT/FP.NORM.ADD.ALKC      ; GO ADD IT IN
                                ;12600
                                ;12601 FP.NORM.DBL.11:
                                ;12602          ;1---------------------------------;
U 0695, 0884,22F7,0030,C047,005A,7 ;12603       R[TEMP3]_(M[TEMP2] RB).RR.9     ; SAVE LOW ORDER 9 BITS AS START OF PACK
                                ;12604
                                ;12605          ;---------------------------------;
                                ;12606          R[TEMP3]_M[TEMP3].RR.16,        ; PACK UP AND FINISH
U 05A7, 0884,33B7,0010,C047,0069,1 ;12607       NEXT/FP.NORM.SNG.20             ;
```

```
                                ;12608 =00
                                ;12609 FP.NORM.DBL.50:
                                ;12610          ;00-------------------------------; ANSWER IS ZERO
                                ;12611          R[TEMP2]_0,                       ; SET FIRST HALF TO ZERO
U 05AC, 0C84,05B7,00B0,8047,0000,1 ;12612       RETURN [+1]                       ; SECOND HALF ALREADY IS
                                ;12613
                                ;12614          ;01-------------------------------;
                                ;12615          M[TEMP3]_(MB R[ZERO]).RR.P,       ; EXTRACT THE FRACTION
U 05AD, 0886,3137,003D,8047,046B,D ;12616       PUSH,NEXT/FP.T3TOT2               ; MOVE T3 TO T2 AND 0 OUT T3
                                ;12617
                                ;12618          ;10-------------------------------;
                                ;12619          M[TEMP5]_MB-ZLIT0[20],            ; SUB 32 TO COUNTERACT PLATCH
                                ;12620          SIZE[WORD],                       ;
U 05AE, 0D86,5C10,0011,0047,0069,5 ;12621       NEXT/FP.NORM.DBL.11               ; JOIN COMMON ROUTINE
                                ;12622 =
                                ;12623 FX.NORM.ADD.ALKC:                          ; USED FOR G H FLOAT
                                ;12624 FP.NORM.ADD.ALKC:                          ;
                                ;12625          ;-------------------------------;
                                ;12626          M[TEMP5]_MB+R[ZERO]+ALKC,         ; ADD ALKC TO TEMP5 (EXPONENT)
                                ;12627          SIZE[WORD],                       ;
U 05CD, 0886,5041,009D,8047,0000,1 ;12628       RETURN [+1]                       ;
                                ;12629
                                ;12630 FX.TESTPSL:
                                ;12631 FP.TESTPSL:
                                ;12632          ;-------------------------------;
U 05CE, 0486,0036,4030,0087,005D,1 ;12633       M[TEMP0]_PSL                      ; GET THE PSL TO TEST FOR FU BIT SET
                                ;12634
                                ;12635          ;-------------------------------;
                                ;12636          WB_M[TEMP0].AND.ZLIT0[40],        ; WAS FU SET
U 05D1, 0D80,0C12,0A32,0047,0069,6 ;12637       WX.EQ.0?                          ;
                                ;12638
                                ;12639 =0
                                ;12640          ;0--------------------------------; FU BIT SET SO FAULT TIME
                                ;12641          VA_M[VA]-ZLIT0[4],                ; DECREMENT THE VA FOR POLYF,POLYD
U 0696, 0549,BC10,0030,24A7,00FF,D ;12642       SET FLAG1,NEXT/IE.FLUN.FAULT      ; SET FLAG1 FOR POLYD FPD RTN
                                ;12643
                                ;12644          ;1--------------------------------; FU BIT CLEAR SO IGNORE
                                ;12645          R[TEMP3]_0,                       ; ZERO OUT SECOND HALF
U 0697, 0884,05B7,00B0,C047,0000,1 ;12646       RETURN [+1]                       ; RETURN TO CALLING ROUTINE
```

```
;12647 .TOC "  Floating point and CRC          : ADD.SIGN ROUTINES "
;12648
;12649 ;*****************************************************************************
;12650 ;
;12651 ;               FP.ADD.SIGNOP1          - (src1 > src2)
;12652 ;
;12653 ;
;12654 ;               FP.ADDD.SIGNOP1         - (src1 > src2) entry double precision
;12655 ;
;12656 ;
;12657 ;               FP.ADD.SIGNOP2          - (src1 < src2)
;12658 ;
;12659 ;                       Input
;12660 ;                                       Temp1           fraction (single precision)
;12661 ;                                                       smaller value
;12662 ;
;12663 ;                                       MDR (bits 0-31)         frction
;12664 ;                                       Temp0 (bits 32-63)      of smaller value
;12665 ;                                                               double precision
;12666 ;
;12667 ;                                       Temp4 (bits 32-63)      if entry FP.ADDD.SIGNOP1
;12668 ;                                       Flag0                   src1 sign flag
;12669 ;                                           (ADD1(FLAG0))
;12670 ;                                       Flag1                   src2 sign flag
;12671 ;                                           (ADD2(FLAG1))
;12672 ;
;12673 ;               Resources:
;12674 ;                                       D                       sign of answer
;12675 ;                                       Flag5                   same sign flag
;12676 ;                                           (SAMESIGN(FLAG4))
;12677 ;
;12678 ;                       Operation:
;12679 ;                                       Enter routines branching on the sign
;12680 ;                                       of the two operands
;12681 ;                                       Where src1>
;12682 ;                                       the sign is the sign of src1 and the src2
;12683 ;                                       value is complemented if the signs are
;12684 ;                                       different the SAMESIGN(FLAG4) is cleared
;12685 ;
;12686 ;                                       where src2 is >
;12687 ;                                       the sign is the sign of src2 and the src1
;12688 ;                                       value is complemented if the signs are unlike
;12689 ;
;12690 ;*****************************************************************************
```

```
                                ;12691 FP.ADD.SIGNOP1:
                                ;12692          ;--------------------------------;
                                ;12693          R[TEMP0] M[TEMP4],              ; SMALLER VALUE TO WORKING TEMP
U 0607, 0884,4592,4530,0047,005B,0 ;12694       ADD2(FLAG1) ADD1(FLAG0)?        ; SIGN OF OPERATION ?
                                ;12695
                                ;12696 =00
                                ;12697 FP.ADD.SIGNOP1:
                                ;12698          ;00--------------------------------; SRC2+,SRC1+
                                ;12699          D_ZLIT0[0],                     ; SET SIGN POSITIVE IN D
U 05B0, 0560,0C37,20B0,0047,0000,3 ;12700       SET SAMESIGN(FLAG4),RETURN [+3] ;
                                ;12701
                                ;12702          ;01--------------------------------; SRC2+,SRC1-
                                ;12703          D_ZLIT12[8],                    ; SET SIGN NEGATIVE IN D
                                ;12704          CLEAR SAMESIGN(FLAG4),          ; CLEAR FLAG
                                ;12705          IR<5>?,                         ; SINGLE OR DOUBLE
U 05B1, 0120,0D77,28F0,4047,0069,8 ;12706       NEXT/FP.ADD.SIGN2               ; DO 2S COMPLEMENT
                                ;12707
                                ;12708          ;10--------------------------------; SRC2-,SRC1+
                                ;12709          D_ZLIT0[0],                     ; SET SIGN POSITIVE IN D
                                ;12710          CLEAR SAMESIGN(FLAG4),          ;
                                ;12711          IR<5>?,                         ; DOUBLE OR SINGLE
U 05B2, 0520,0C37,28F0,0047,0069,8 ;12712       NEXT/FP.ADD.SIGN2               ; DO 2S COMPLEMENT
                                ;12713
                                ;12714          ;11--------------------------------; SRC2-, SRC1-
                                ;12715          D_ZLIT12[8],                    ; SET SIGN NEGATIVE IN D
U 05B3, 0160,0D77,20B0,4047,0000,3 ;12716       SET SAMESIGN(FLAG4),RETURN [+3] ;
                                ;12717
                                ;12718 =00
                                ;12719 FP.ADD.SIGNOP2:
                                ;12720          ;00--------------------------------; SRC2+,SRC1+
U 05B4, 0D80,0C37,20B0,0047,0000,3 ;12721       D_ZLIT0[0],RETURN [+3]          ; SET SIGN POSITIVE IN D
                                ;12722
                                ;12723          ;01--------------------------------; SRC2+,SRC1-
                                ;12724          D_ZLIT0[0],                     ; SET SIGN POSITIVE IN D
                                ;12725          IR<5>?,                         ; SINGLE OR DOUBLE
U 05B5, 0580,0C37,28F0,0047,0069,8 ;12726       NEXT/FP.ADD.SIGN2               ; DO 2S COMPLEMENT
                                ;12727
                                ;12728          ;10--------------------------------; SRC2-,SRC1+
                                ;12729          D_ZLIT12[8],                    ; SET SIGN NEGATIVE IN D
                                ;12730          IR<5>?,                         ; DOUBLE OR SINGLE
U 05B6, 0180,0D77,28F0,4047,0069,8 ;12731       NEXT/FP.ADD.SIGN2               ; DO 2S COMPLEMENT
                                ;12732
                                ;12733          ;11--------------------------------; SRC2-, SRC1-
U 05B7, 0980,0D77,20B0,4047,0000,3 ;12734       D_ZLIT12[8],RETURN [+3]         ; SET SIGN NEGATIVE IN D
                                ;12735 =0
                                ;12736 FP.ADD.SIGN2:
                                ;12737          ;0---------------------------------; FLOAT SINGLE
U 0698, 0086,1003,00BD,8047,0000,3 ;12738       M[TEMP1]_-MB,RETURN [+3]        ; 2S COMPLEMENT
                                ;12739
                                ;12740          ;1---------------------------------; FLOATING DOUBLE
U 0699, 0886,0003,003D,8047,0060,8 ;12741       M[TEMP0]_-MB                    ;
                                ;12742
                                ;12743          ;--------------------------------;
                                ;12744          MDR_R[ZERO]-M[MDR]-ALKC,        ; 2S COMPLEMENT SIGNIFICANT HALF
U 060B, 0081,2043,00BD,8467,0000,3 ;12745       RETURN [+3]                     ;
```

```
;12746 .TOC "  Floating point and CRC          : FP.MULF23 "
;12747
;12748 ;*************************************************************************
;12749 ;
;12750 ;       44      MULF2   mulr.rf,prod.mf
;12751 ;
;12752 ;       45      MULF3   mulr.rf,muld.rf,proj.wf
;12753 ;
;12754 ;
;12755 ;               Input:
;12756 ;                                       Q               mulr.rf
;12757 ;                                       MDR             muld.rf/prod.mf
;12758 ;
;12759 ;               Resources:
;12760 ;                       for mulr
;12761 ;                               Temp2,Q         fraction
;12762 ;                               Temp5           exponent
;12763 ;                               Flag2           sign of operand
;12764 ;                                  (MUL1(FLAG2))
;12765 ;
;12766 ;                       for muld
;12767 ;                               Temp0           fraction
;12768 ;                               RTemp8          exponent
;12769 ;                               Flag3           sign of operand
;12770 ;                                  (MUL2(FLAG3))
;12771 ;
;12772 ;                       Flag1                   flag set if either
;12773 ;                         (MOPZERO(FLAG1))      operand zero
;12774 ;                                               all flags are zeroed
;12775 ;                                               if an operand is zero
;12776 ;
;12777 ;                               Platch          to shift operand
;12778 ;                               Stepcounter     for multiply iteration count
;12779 ;                               D,Q             for multiply operation
;12780 ;
;12781 ;
;12782 ;       Subroutines:
;12783 ;               FP.MULFOP1              parse mulr.rf
;12 84 ;               FP.MULFOP2              parse muld.rf/prod.wf
;12.85 ;               FP.NORM.MUDI            normalize round pack product
;12786 ;
;12787 ;       Exit:
;12788 ;               FP.WRITE23
;12789 ;
;12790 ;       Output:
;12791 ;               Temp2                   packed product
;12792 ;               Flag2                   overflow flag
;12793 ;                  (OVER(FLAG2))                set if appropriate
;12794 ;               Flag3                   underflow flag
;12795 ;                  (UNDER(FLAG3))       set if appropriate
;12796 ;
;12797 ;       Operation:
;12798 ;               Both operands are validated and parsed
;12799 ;               the exponents are added
```

```
                                ;12800 ;                       If either operand is zero the product is zero
                                ;12801 ;                       and an exit is effected
                                ;12802 ;                       otherwise, the stepcounter is set to 14
                                ;12803 ;                       the two operands are shifted right 5 places
                                ;12804 ;                       and the multiplier (Temp2) also placed in Q
                                ;12805 ;
                                ;12806 ;                       The multiplication is performed 2 bits at a time
                                ;12807 ;                       ending with 26 bits of product in D
                                ;12808 ;                       the binary point is at bit 25 and the
                                ;12809 ;                       "fudge" factor  for normalization is
                                ;12810 ;                       99-hex (80 for the extra bias 19 for platch
                                ;12811 ;                       and the bp of 25)
                                ;12812 ;
                                ;12813 ;                       Entry to the normalization routine is made where
                                ;12814 ;                       the XOR of the two sign determines the
                                ;12815 ;                       sign of the product
                                ;12816 ;
                                ;12817 ;                       Upon return a branching exit is made to
                                ;12818 ;                       differentiate between MULF2 and MULF3
                                ;12819 ;
                                ;12820 ;       **********FPA ENTRY AND INTERFACE**********
                                ;12821 ;       MULF2   (OS.RED) - FI.ADDF2.REG/FI.ADDF2.MEM
                                ;12822 ;       MULF3   (OS.RED) - FI.ADDF3.REG/FI.ADDF3.MEM
                                ;12823 ;
                                ;12824 ;*************************************************************************
                                ;12825
                                ;12826 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;12827 =000
                                ;12828         ;000-----------------------------; MULR (SRC1)  IS ZERO
U 0080, 0480,0036,4030,0047,0008,2  ;12829         NEXT/FP.MULF.10                 ; WASTE WORD BUT
                                ;12830
                                ;12831 FP.MULF23:                               ; **********************
                                ;12832         ;001-----------------------------; START OF MULF2/3
                                ;12833         M[TEMP2] Q,                     ; Q TO AN MBUS SOURCE
U 0081, 0886,203A,403D,8047,0463,A  ;12834         PUSH,NEXT/FP.MULFOP1            ; PARSE MULR
                                ;12835
                                ;12836 FP.MULF.10:
                                ;12837         ;010-----------------------------;
                                ;12838         R[TEMP8] EXP(M[MDR]),           ; EXTRACT EXPONENT
                                ;12839         FRO.FLTZ?,                      ; TEST VALIDITY OF MULD
U 0082, 0C85,2437,0AB2,0047,0C5B,C 323*  ;12840         PUSH,NEXT/FP.MULFOP2            ;
                                ;12841
                                ;12842         ;011-----------------------------;
                                ;12843         M[TEMP5] MB+R[TEMP8],           ; ADD EXPONENTS
                                ;12844         SIZE[WORD],                     ;
                                ;12845         MOPZERO(FLAG1)?,                ; EITHER OPERAND ZERO?
U 0083, 0C86,5001,05D2,0047,045D,4  ;12846         PUSH,NEXT/FP.MULF.200           ; PREPARE FOR THE MULTIPLICATION
```

```
                                ;12847 FP.MULF.100:
                                ;12848          ;100------------------------------; THE MULTIPLICATION
                                ;12849          MULFAST+ CAND IN R[TEMP0],       ; STEPCTR IS 14 AT START
                                ;12850          SIZE[LONG],                      ;
                                ;12851          DBZ STEPC?,                      ; DECREMENT & BRANCH TILL DONE
U 0084, 0480,0027,9320,0047,0008,4 ;12852       NEXT/FP.MULF.100                 ; Q HAS MULTIPLIER TO START
                                ;12853                                           ; RBUS HAS MULTIPLICAND
                                ;12854                                           ; PRODUCT IN D AT END
                                ;12855
                                ;12856          ;101------------------------------; ALL DONE THE MULTIPLY
                                ;12857          M[TEMP5] MB-ZLIT0[99],           ; ADD FACTORS
U 0085, 0186,5C10,0014,C847,0008,6 ;12858       SIZE[WORD]                       ;   19 FOR PLATCH
                                ;12859                                           ; AND BIT 25 POSITION OF BP
                                ;12860                                           ; =99(HEX) WITH 80 FOR EXTRA BIAS
                                ;12861
                                ;12862          ;110------------------------------;
                                ;12863          M[TEMP2] D,                      ; MOVE ANS TO TEMP FOR NORM
                                ;12864          MUL1(FLAG2) XOR MUL2(FLAG3)?,    ; XOR OF23 = PRODUCT SIGN
                                ;12865          PUSH,                            ; FLAG1 IS CONSTRAINED OUT
U 0086, 0C86,25B2,4570,0047,0450,A ;12866       NEXT/FP.NORM.MUDI                ;
                                ;12867
                                ;12868          ;110------------------------------;
                                ;12869          IR<2-0>?,                        ; 2/3 OPERANDS
U 0087, 0080,0036,4670,0047,0050,6 ;12870       NEXT/FP.WRITE23                  ; DO WRITE OR GET THIRD OPERAND
                                ;12871
                                ;12872 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;12873 =0*
                                ;12874 FP.MULF.200:
                                ;12875          ;0*------------------------------; MOPZERO(FLAG1) NOT SET
                                ;12876          PL [5],                          ; SET PLATCH TO 5 FOR MOVING OPERANDS
                                ;12877          STEPC 14.,                       ; SET STEP TO 14 (26 BITS + 1 SETUP)
U 05D4, 0DB0,0EF6,4030,2847,0061,0 ;12878       NEXT/FP.MULF.250                 ;
                                ;12879
                                ;12880          ;1*------------------------------; MOPZERO(FLAG1) IS SET
                                ;12881          R[TEMP2]_FLAGS_0,                ; ANSWER IS 0,ZERO FLAGS TOO
                                ;12882          IR<2-0>?,                        ; 2/3 OPERAND WRITE
U 05D6, 0C84,05BE,0670,8307,0050,6 ;12883       NEXT/FP.WRITE23                  ;
                                ;12884
                                ;12885 FP.MULF.250:
                                ;12886          ;-------------------------------;
U 0610, 0C86,20F7,1030,0047,0063,8 ;12887       M[TEMP2]_Q_MB.ASR.P              ; SHIFT MUL RIGHT 5 (PLATCH)
                                ;12888
                                ;12889
                                ;12890          ;-------------------------------;
                                ;12891          M[TEMP0]_MB.ASR.P,               ; SHIFT MUL RIGHT 5 (PLATCH)
U 0638, 0886,00F7,00B0,0047,0000,1 ;12892       RETURN [+1]                      ;
```

```
;12893 .TOC "  Floating point and CRC          : FP.MULFOP1 FP.MULFOP2 "
;12894
;12895 ;*************************************************************************
;12896 ;
;12897 ;       SINGLE PRECISION OPERAND PARSERS
;12898 ;
;12899 ;               FP.MULFOP1
;12900 ;
;12901 ;
;12902 ;                       Input:
;12903 ;                               Temp2           packed operand
;12904 ;                               Flag2           clear
;12905 ;                                  (MUL1(FLAG2))
;12906 ;                               Flag1           clear
;12907 ;                                  (MOPZERO(FLAG1))
;12908 ;
;12909 ;                       Resources:
;12910 ;                               Temp2,Q         fraction unpacked
;12911 ;                                                 binary point at bit 30
;12912 ;                                                 overflow and hidden bit 31-30
;12913 ;                               Temp5           exponent
;12914 ;                               Flag2           sign of operand
;12915 ;                                  (MUL1(FLAG2))
;12916 ;                               Flag1           set if operand zero
;12917 ;                                  (MOPZERO(FLAG1))
;12918 ;
;12919 ;                       Exit
;12920 ;                               Return +1       if operand valid and >,< 0
;12921 ;                               Return -1       if operand =0
;12922 ;                               IE.OPER.FAULT   if reserved operand
;12923 ;
;12924 ;
;12925 ;               FP.MULFOP2
;12926 ;
;12927 ;                       Input:
;12928 ;                               MDR             packed operand
;12929 ;                               Flag3           clear
;12930 ;                                  (MUL2(FLAG3))
;12931 ;                               Flag1           maybe either way
;12932 ;                                  (MOPZERO(FLAG1))
;12933 ;
;12934 ;                       Resources:
;12935 ;                               Temp0           unpacked fraction
;12936 ;                               Flag3           sign of operand
;12937 ;                                 (MUL2(FLAG3))
;12938 ;                               Flag1           if either operand 0 is set
;12939 ;                                  (MOPZERO(FLAG1))
;12940 ;
;12941 ;                       Exit:
;12942 ;                               Return +1       if valid >,<,=0
;12943 ;                               IE.OPER.FAULT   if reserved operand
;12944 ;
;12945 ;
;12946 ;*************************************************************************
```

```
                                        ;12947 FP.MULFOP1:
                                        ;12948          ;-------------------------------;
                                        ;12949          R[TEMP5]_EXP(M[TEMP2]),         ; PARSE OPERAND 1
U 063A, 0C84,2437,0AB1,4047,085B,8 323* ;12950          FRO.FLTZ?                       ; TEST VALIDITY
                                        ;12951
                                        ;12952 =00
                                        ;12953          ;00-------------------------------; RESERVED OPERAND
U 05B8, 0C80,0036,4030,0047,00FF,8      ;12954          NEXT/IE.OPER.FAULT              ;
                                        ;12955
                                        ;12956          ;01-------------------------------; SRC1 < 0
                                        ;12957          M[TEMP2]_Q_UNPACK(MB R[ZERO]),  ; UNPACK THE FRACTION
                                        ;12958          SET MUL1(FLAG2),                ; MULTIPLIER DIVISOR
U 05B9, 0456,2677,10BD,8047,0000,1      ;12959          RETURN [+1]                     ;
                                        ;12960
                                        ;12961          ;10-------------------------------; SRC1 = 0
                                        ;12962          R[TEMP2]_Q_0,                   ; SET UP ZERO
                                        ;12963          SET MOPZERO(FLAG1),             ; SET FLAG TO INDICATE ZERO
U 05BA, 084C,05B7,10B0,8047,03FF,F      ;12964          RETURN [-1]                     ; NOTE DIFFERENT RETURN ****
                                        ;12965
                                        ;12966          ;11-------------------------------; SRC1 > 0
                                        ;12967          M[TEMP2]_Q_UNPACK(MB R[ZERO]),  ; UNPACK THE FRACTION
                                        ;12968          CLEAR MUL1(FLAG2),              ;
U 05BB, 0C16,2677,10BD,8047,0000,1      ;12969          RETURN [+1]                     ;
                                        ;12970
                                        ;12971 =00
                                        ;12972 FP.MULFOP2:
                                        ;12973          ;00-------------------------------; RESERVED OPERAND
U 05BC, 0C80,0036,4030,0047,00FF,8      ;12974          NEXT/IE.OPER.FAULT              ;
                                        ;12975
                                        ;12976          ;01-------------------------------; SRC2 < 0
                                        ;12977          MTEMP0_UNPACK(MDR R[ZERO]),     ; UNPACK THE FRACTION
                                        ;12978          SET MUL2(FLAG3),                ; SET MULTIPLICAND DIVIDEND FLAG
U 05BD, 0C5F,2677,00BD,8047,0000,1      ;12979          RETURN [+1]                     ;
                                        ;12980
                                        ;12981          ;10-------------------------------; SRC2 = 0
                                        ;12982          R[TEMP0]_0,                     ; ZERO OUT THE OPERAND
                                        ;12983          MDR_0,                          ; ALSO THE PACKED FORM
                                        ;12984          SET MOPZERO(FLAG1),             ;
U 05BE, 084C,05B7,00B0,04E7,0000,1      ;12985          RETURN [+1]                     ;
                                        ;12986
                                        ;12987          ;11-------------------------------; SRC2 > 0
                                        ;12988          MTEMP0_UNPACK(MDR R[ZERO]),     ; UNPACK THE FRACTION
                                        ;12989          CLEAR MUL2(FLAG3),              ;
U 05BF, 041F,2677,00BD,8047,0000,1      ;12990          RETURN [+1]                     ;
```

```
;12991 .TOC "  Floating point and CRC          : FP.DIVF23 "
;12992
;12993 ;*************************************************************************************
;12994 ;
;12995 ;       46      DIVF2   divr.rf,quo.mf
;12996 ;
;12997 ;       47      DIVF3   divr.rf,divd.rf,quo.wf
;12998 ;
;12999 ;                       Input:
;13000 ;                               Q               divisor
;13001 ;                               MDR             dividend
;13002 ;                       Resources:
;13003 ;                               for divisor
;13004 ;                               Temp2,Q         fraction
;13005 ;                               Temp5           exponent
;13006 ;                               Flag2           sign of divisor
;13007 ;                                  (MUL1(FLAG2))
;13008 ;
;13009 ;                               for dividend
;13010 ;                               Temp0,MDR       fraction
;13011 ;                               RTemp8          exponent
;13012 ;                               Flag3           sign of dividend
;13013 ;                                  (MUL2(FLAG3))
;13014 ;
;13015 ;                               Flag1           set if either operand 0
;13016 ;                                  (MOPZERO(FLAG1))
;13017 ;                               Stepcounter     division loop control
;13018 ;                               D,Q             for division operation
;13019 ;                       Subroutines:
;13020 ;                               FP.MULFOP1      parse divisor
;13021 ;                               FP.MULFOP2      parse dividend
;13022 ;                               FP.NORM.MUDI    normalize,round, pack quotient
;13023 ;                       Exits:
;13024 ;                               FP.WRITE23      to write out quotient
;13025 ;                       Operation:
;13026 ;                               The divisor is parsed, validated. if 0
;13027 ;                               a fault exit is made to IE.FLDBZ.FAULT
;13028 ;                               Else the dividend is parsed,validated
;13029 ;                               if 0 the quotient is 0  exit to FP.WRITE23
;13030 ;                               Else D and Q are set up to perform the division
;13031 ;                               2 bits at a time - 26 bit quotient(Q)
;13032 ;                               Exponents are subtracted and a "fudge" factor
;13033 ;                               of +67-hex added ( -19 for the binary point
;13034 ;                               at bit 25, and +80 for the bias)
;13035 ;                               Normalizing,rounding packing done entering
;13036 ;                               at FP.NORM.MUDI to determine sign of quotient
;13037 ;                               by the XOR of the operand signs Upon return exit
;13038 ;                               branches to differentiate between DIVF2,DIVF3
;13039 ;
;13040 ;       **********FPA ENTRY AND INTERFACE**********
;13041 ;       DIVF2   (OS.RED) - FI.ADDF2.REG/FI.ADDF2.MEM
;13042 ;       DIVF3   (OS.RED) - FI.ADDF3.REG/FI.ADDF3.MEM
;13043 ;
;13044 ;*************************************************************************************
```

```
                                ;13045 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;13046 =00
                                ;13047          ;00-----------------------------; DIVISOR IS ZERO
U 0094, 0480,0036,4030,0047,00FF,C ;13048       NEXT/IE.FLDBZ.FAULT             ; OFF TO FAULT EXIT
                                ;13049
                                ;13050 FP.DIVF23:                               ; **********************
                                ;13051          ;01-----------------------------; MAIN ENTRY
                                ;13052          M[TEMP2] Q,                     ; GET DIVISOR (SRC1) TO MBUS SOURCE
U 0095, 0886,203A,403D,8047,0463,A ;13053       PUSH,NEXT/FP.MULFOP1            ; TEST VALIDITY OF OPERAND
                                ;13054
                                ;13055          ;10-----------------------------; RETURN IF DIVISOR NE 0
                                ;13056          R[TEMP8] EXP(M[MDR]),           ; PARSE DIVIDEND (SRC2)
                                ;13057          FRO.FLTZ?,                      ; TEST VALIDITY
U 0096, 0C85,2437,0AB2,0047,0C5B,C 323* ;13058  PUSH,NEXT/FP.MULFOP2             ;
                                ;13059
                                ;13060          ;11-----------------------------;
                                ;13061          M[TEMP5] R[TEMP8]-MB,           ; SUB EXPONENTS
                                ;13062          SIZE[WORD],                     ;
                                ;13063          MDR_0,                          ;
U 0097, 0886,5003,05D2,04E7,005D,8 ;13064       MOPZERO(FLAG1)?                 ;
                                ;13065
                                ;13066 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;13067 =0*
                                ;13068          ;0*-----------------------------;
                                ;13069          Q_M[MDR] D_R[TEMP0],            ; SET UP DIVIDEND FOR DIVISION
U 05D8, 04B1,2005,7030,0047,005C,0 ;13070       STEPC_14.,NEXT/FP.DIVF.40       ; DO DIVIDE
                                ;13071
                                ;13072          ;1*-----------------------------;
                                ;13073          R[TEMP2]_FLAGS_0,               ; ANS IS ZERO
                                ;13074          IR<2-0>?,                       ;
U 05DA, 0C84,05BE,0670,8307,0050,6 ;13075       NEXT/FP.WRITE23                 ; GO WRITE OUT
                                ;13076 =00
                                ;13077 FP.DIVF.40:
                                ;13078          ;00-----------------------------; THE DIVISION
                                ;13079          DIVFAST+ SOR IN R[TEMP2],       ;
                                ;13080          SIZE[LONG],DBZ STEPC?,          ; STEPCOUNTER 14 TO START
U 05C0, 0480,0026,C320,8047,005C,0 ;13081       NEXT/FP.DIVF.40                 ;
                                ;13082
                                ;13083          ;01-----------------------------; PREPARE EXPONENT
                                ;13084          M[TEMP5] MB+?LITO[67],          ; -19(HEX)PLATCH &BP
U 05C1, 0586,5C11,0013,3847,0069,A ;13085       SIZE[WORD]                      ; +80 BIAS = 67 TOTAL
                                ;13086 =
                                ;13087 =0
                                ;13088          ;0-----------------------------;
                                ;13089          M[TEMP2] Q,                     ; PUT INTO NORM REG
                                ;13090          MUL1(FLAG2) XOR MUL2(FLAG3)?,   ; SIGN OF QUO TEST
U 069A, 0C86,203A,457D,8047,0450,A ;13091       PUSH,NEXT/FP.NORM.MUDI          ;
                                ;13092
                                ;13093          ;1-----------------------------;
                                ;13094          IR<2-0>?,                       ; PREPARE FOR WRITE OUT
U 069B, 0080,0036,4670,0047,0050,6 ;13095       NEXT/FP.WRITE23                 ;
```

```
;13096 .TOC "  Floating point and CRC          : FP.ADDD2 FP.SUBD2 "
;13097
;13098 ;*****************************************************************************
;13099 ;
;13100 ;       60      ADDD2   add.rd,sum.md
;13101 ;
;13102 ;       62      SUBD2   sub.rd,dif.md
;13103 ;
;13104 ;
;13105 ;                       Input:
;13106 ;                               Temp1           src1 (bits 0-31)
;13107 ;                               MDR             (bits 32-63)
;13108 ;
;13109 ;
;13110 ;                       Resources:
;13111 ;                          for src1
;13112 ;                               Temp2,Q         fraction (bits 0-31)
;13113 ;                               Temp3                    (bits 32-63)
;13114 ;                               Temp5           exponent
;13115 ;                               Flag0           sign of operand
;13116 ;                                  (ADD1(FLAG0))
;13117 ;                               Flag1           SUBD inst till 1st operand
;13118 ;                                  (SUB(FLAG1))  parsed
;13119 ;
;13120 ;                       Subroutines:
;13121 ;                               FP.ADDDOP1      parse operand
;13122 ;                               OS.DMUD         get src2
;13123 ;
;13124 ;                       Exit:
;13125 ;                                       FP.WRITE.SEC    write second part of answer
;13126 ;
;13127 ;                       Operation:
;13128 ;                               The entry for the SUBD2 instruction sets the
;13129 ;                               flag and falls into common code with ADDD2
;13130 ;                               The src1 operand is parsed and validated
;13131 ;                               In the  case of SUBD2 this results in the
;13132 ;                               sign being inverted. The second operand
;13133 ;                               is obtained via a call to OS.DMOD
;13134 ;                               Return is through IRDROM to FP.ADDD.20
;13135 ;                               where the operation continues.
;13136 ;                               Upon return bits 0-31 of the answer are written
;13137 ;                               and FP.WRITE.SEC is entered to finish up
;13138 ;
;13139 ;       **********FPA ENTRY AND INTERFACE**********
;13140 ;                       (OS.FIDRED) - FI.ADDD2.REG/FI.ADDD2.MEM
;13141 ;
;13142 ;*****************************************************************************
```

```
                                ;13143 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;13144 FP.SUBD2:                        ; **********
                                ;13145          ;-----------------------;
U 0356, 0448,0036,4030,0047,000A,8      ;13146          SET SUB(FLAG1)          ; SET FLAG INDICATING SUBD INSTRUCTION
                                ;13147
                                ;13148 =00
                                ;13149 FP.ADDD2:                        ; **********
                                ;13150          ;00---------------------;
                                ;13151          R[TEMP5]_EXP(M[TEMP1]),         ; PARSE ADD1
                                ;13152          FRO.FLTZ?,              ; TEST VALIDITY OF OPERAND
U 00A8, 0C84,1437,0AB1,4047,0C5C,4 323*  ;13153          PUSH,NEXT/FP.ADDDOP1    ;
                                ;13154
                                ;13155          ;01---------------------;
                                ;13156          M[TEMP2]_Q,             ; SAVE BITS 0-31 OF FRACTION OF ADD1
                                ;13157          PUSH,                   ;
                                ;13158          CLEAR SUB(FLAG1),       ; BE SURE SUBD FLAG IS CLEAR
                                ;13159          LOD INC BRA?,           ; GET ADD2
U 00A9, 000E,203A,41BD,8047,0419,0      ;13160          NEXT/OS.DMOD            ; RETURN IS FIRST THROUGH IRDROM
                                ;13161                                  ; TO FP.ADDD.20
                                ;13162                                  ; FINAL RETURN HERE WHEN HAVE ANSWER
                                ;13163
                                ;13164          ;10---------------------;
                                ;13165          R[DST.R]_M[TEMP2],      ; FIRST INSTRUCTION OF WRITE
                                ;13166          CCOP1,                  ; SET CONDITION CODES
                                ;13167          WRITE NOTREG,SIZE[IDEP],        ; ANS IS IN TEMP2 TEMP3
                                ;13168          SET MM.NOINT,           ; NO INTERRUPTS ALLOWED BETWEEN WRITE
U 00AA, 0064,2592,403C,4DDA,0056,F      ;13169          NEXT/FP.WRITE.SEC       ; WRITE SECOND HALF
                                ;13170 =
```

```
;13171 .TOC "  Floating point and CRC          : FP.ADDD3 FP.SUBD3 "
;13172
;13173 ;*****************************************************************************
;13174 ;
;13175 ;
;13176 ;       61      ADDD3   add1.rd,add2.rd,sum.wd
;13177 ;
;13178 ;       63      SUBD3   sub.rd,min.rd,dif.wd
;13179 ;
;13180 ;                       Input:
;13181 ;                               Temp1           src1 (bits 0-31)
;13182 ;                               MDR             (bits 32-63)
;13183 ;
;13184 ;
;13185 ;                       Resources:
;13186 ;                          for src1
;13187 ;                               Temp2,Q         fraction (bits 0-31)
;13188 ;                               Temp3                    (bits 32-63)
;13189 ;                               Temp5           exponent
;13190 ;                               Flag0           sign of operand
;13191 ;                                  (ADD1(FLAG0))
;13192 ;                               Flag1           SUBD inst till 1st operand
;13193 ;                                  (SUB(FLAG1))  parsed
;13194 ;
;13195 ;                       Subroutines:
;13196 ;                               FP.ADDDOP1      parse operand
;13197 ;                               OS.DRED         get src2
;13198 ;                               OS.WRT1         get sum.wd/dif.wd
;13199 ;
;13200 ;                       Exit:
;13201 ;                               FP.WRITE.SEC    write second part of answer
;13202 ;
;13203 ;                       Operation:
;13204 ;                               The entry for the SUBD3 instruction sets the
;13205 ;                               flag and falls into common code with ADDD3
;13206 ;                               The src1 operand is parsed and validated
;13207 ;                               In the  case of SUBD3 this results in the
;13208 ;                               sign being inverted. The second operand
;13209 ;                               is obtained via a call to OS.DRED
;13210 ;                               Return is through IRDROM to FP.ADDD.20
;13211 ;                               where the operation continues.
;13212 ;                               Upon return OS.WRT1 is called to get third
;13213 ;                               operand  upon return (+1) bits 0-31 are written
;13214 ;                               exit is to FP.WRITE.SEC to finish up
;13215 ;       **********FPA ENTRY AND INTERFACE**********
;13216 ;               (OS.FIDRED) - FI.ADDD3.REG/FI.ADDD3.MEM
;13217 ;
;13218 ;*****************************************************************************
```

```
                                ;13219 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;13220 FP.SUBD3:                          ; **********************
                                ;13221          ;-------------------------------;
U 0357, 0C48,0036,4030,0047,000B,4 ;13222       SET SUB(FLAG1)            ; SET FLAG INDICATING SUBD INSTRUCTION
                                ;13223
                                ;13224 =00
                                ;13225 FP.ADDD3:                          ; **********************
                                ;13226          ;00-------------------------------;
                                ;13227          R[TEMP5]_EXP(M[TEMP1]),   ; PARSE ADD1
                                ;13228          FRO.FLTZ?,                ; TEST VALIDITY OF OPERAND
U 00B4, 0C84,1437,0AB1,4047,0C5C,4 323* ;13229  PUSH,NEXT/FP.ADDDOP1       ;
                                ;13230
                                ;13231          ;01-------------------------------;
                                ;13232          M[TEMP2]_Q,               ; SAVE BITS 0-31 OF ADD1 SRC1
                                ;13233          PUSH,                     ;
                                ;13234          CLEAR SUB(FLAG1),         ; CLEAR SUBD FLAG
                                ;13235          LOD INC BRA?,             ; GET ADD2
U 00B5, 080E,203A,41BD,8047,0418,0 ;13236       NEXT/OS.DRED              ; RETURN THROUGH IRDROM TO FP.ADDD.20
                                ;13237                                    ; FINAL RETURN HERE TO GET THIRD OPERAND
                                ;13238
                                ;13239          ;10-------------------------------;
                                ;13240          LOD INC BRA?,             ; GET THIRD OPERAND TO WRITE
U 00B6, 0C80,0036,41B0,0047,0414,0 ;13241       PUSH,NEXT/OS.WRT1         ;
                                ;13242
                                ;13243          ;11-------------------------------;
                                ;13244          R[DST.R]_M[TEMP2],        ; FIRST INSTRUCTION OF WRITE
                                ;13245          CCOP1,                    ; SET CONDITION CODES
                                ;13246          WRITE NOTREG,SIZE[IDEP],  ;
                                ;13247          SET MM.NOINT,             ; NO INTERRUPTS BETWEEN WRITES
U 00B7, 0064,2592,403C,4DDA,0056,F ;13248       NEXT/FP.WRITE.SEC         ; WRITE SECOND HALF
                                ;13249 =
```

```
;13250 .TOC "  Floating point and CRC          : FP.ADDDOP1 FP.ADDDOP2 "
;13251
;13252 ;*****************************************************************************
;13253 ;
;13254 ;       DOUBLE PRECISION OPERAND PARSERS
;13255 ;
;13256 ;               FP.ADDDOP1
;13257 ;
;13258 ;                       Input:
;13259 ;                               Temp1           src1 (bits 0-31)
;13260 ;                               MDR     (bits 32-63)
;13261 ;                               Temp5           exponent
;13262 ;                               Flag1           set if SUBD2/SUBD3 instruction
;13263 ;                                  (SUB(FLAG1))
;13264 ;
;13265 ;                       Resources:
;13266 ;                               Q               fraction (bits 0-31)
;13267 ;                                               with bits 31-30
;13268 ;                                                 overflow and hidden bits
;13269 ;                               Temp3           fraction (bits 32-63)
;13270 ;                               Flag0           sign of operand
;13271 ;                                  (ADD1(FLAG0))
;13272 ;                               Flag3           set if operand 0
;13273 ;                                  (OPZERO(FLAG3))
;13274 ;                       Exits:
;13275 ;                               Return +1       if operand valid and >,<,=0
;13276 ;                               IE.OPER.FAULT   if reserved operand
;13277 ;
;13278 ;                       **This routine called at FP.ADDDOP1.20 by FP.MULDOP1
;13279 ;
;13280 ;               FP.ADDDOP2
;13281 ;
;13282 ;                       Input:
;13283 ;                               Temp1           src2 (bits 0-31)
;13284 ;                               MDR             (bits 32-63)
;13285 ;                               Temp7           exponent
;13286 ;
;13287 ;                       Resources:
;13288 ;                               Temp1           fraction (bits 0-31)
;13289 ;                               Temp4           (bits 32-63)
;13290 ;                               Flag1           sign of operand
;13291 ;                                  (ADD2(FLAG1))
;13292 ;
;13293 ;                       Subroutines:
;13294 ;                               FP.ZEROTEMP1    to zero temp
;13295 ;
;13296 ;                       Operation:
;13297 ;                               Operand is parsed  if not zero or reserved
;13298 ;                               this involves rotating bits 32-63 and
;13299 ;                               unpacking bits 0-31  the
;13300 ;                               remaining 7 bits that are already a part of
;13301 ;                               bits 0-31 are removed from bits 32-63
;13302 ;
;13303 ;*****************************************************************************
```

```
                                ;13304 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;13305 =00
                                ;13306 FX.ADDHOP1:                              ; ENTRY FOR GH FLOAT
                                ;13307 FP.ADDDOP1:                              ;
                                ;13308          ;00--------------------------; RESERVED OPERAND
U 05C4, 0C80,0036,4030,0047,00FF,8 ;13309       NEXT/IE.OPER.FAULT              ;
                                ;13310
                                ;13311          ;01--------------------------; ADD1 < 0
                                ;13312          R[TEMP3]_M[MDR].RR.16,          ; ROTATE BITS 32-63
                                ;13313          SET ADD1(FLAG0),                ; SET SIGN FLAG
U 05C5, 0845,23B7,0010,C047,005D,C ;13314       NEXT/FP.ADDDOP1.20              ; NOW UNPACK THE FRACTION
                                ;13315
                                ;13316          ;10--------------------------; ADD1 = 0
                                ;13317          R[TEMP3]_Q_0,                   ; SET ADD1 TO 0
                                ;13318          MDR_0,                          ; ALSO THE SECOND HALF
                                ;13319          SET OPZERO(FLAG3),              ; SIGNAL THIS
U 05C6, 045C,05B7,10B0,C4E7,0000,1 ;13320       RETURN [+1]                     ; BACK TO GET SECOND OPERAND
                                ;13321
                                ;13322          ;11--------------------------; ADD1 > 0
                                ;13323          R[TEMP3]_M[MDR].RR.16,          ; ROTATE BITS 32-63
                                ;13324          SUB(FLAG1)?,                    ; WAS THIS A SUBD INSTRUCTION
U 05C7, 0885,23B7,05D0,C047,005D,C ;13325       NEXT/FP.ADDDOP1.20              ; IF SO SIGN GETS FLIPPED
                                ;13326
                                ;13327 =0*
                                ;13328 FP.ADDDOP1.20:
                                ;13329          ;0*--------------------------;
                                ;13330          Q_UNPACK(M[TEMP1] R[TEMP3]),    ; UNPACK GETS BITS 0-31
                                ;13331          SUB(FLAG1)?,                    ; WAS IT SUBD INSTRUCTION (HERE FROM
                                ;13332                                          ; ADD1 BEING NEGATIVE)
U 05DC, 0880,1677,15F0,C047,005E,0 ;13333       NEXT/FP.ADDDOP1.30              ; CONTINUE THE UNPACK
                                ;13334
                                ;13335          ;1*--------------------------;
                                ;13336          Q_UNPACK(M[TEMP1] R[TEMP3]),    ; UNPACK GETS BITS 0-31 OF FRACTION
U 05DE, 0840,1677,1030,C047,005E,0 ;13337       SET ADD1(FLAG0)                 ; HERE ONLY IF SUBD INSTRUCTION AND
                                ;13338                                          ; ADD1 WAS POSITIVE
                                ;13339 =0*
                                ;13340 FP.ADDDOP1.30:
                                ;13341          ;0*--------------------------;
                                ;13342          R[TEMP3]_RB.ASL.7,              ; GETS LOW ORDER BITS 32-63
U 05E0, 0884,0577,00B0,C047,0000,1 ;13343       RETURN [+1]                     ;
                                ;13344
                                ;13345          ;1*--------------------------;
                                ;13346          R[TEMP3]_RB.ASL.7,              ; GETS LOW ORDER BITS 32-63
                                ;13347          CLEAR ADD1(FLAG0),              ; HERE IF SUBD AND ADD1 WAS NEGATIVE
U 05E2, 0004,0577,00B0,C047,0000,1 ;13348       RETURN [+1]                     ;
```

```
                                ;13349 =00
                                ;13350 FP.ADDDOP2:
                                ;13351          ;00-----------------------------; RESERVED OPERAND
U 05C8, 0C80,0036,4030,0047,00FF,8 ;13352       NEXT/IE.OPER.FAULT              ;
                                ;13353
                                ;13354          ;01-----------------------------; ADD2 < 0
                                ;13355          R[TEMP4]_M[MDR].RR.16,          ; ROTATE BITS 32-63
                                ;13356          SET ADD2(FLAG1),                ; SET SIGN FLAG
U 05C9, 0C4D,23B7,0011,0047,0063,F ;13357       NEXT/FP.ADDDOP2.20              ; CONTINUE PROCESSING
                                ;13358
                                ;13359          ;10-----------------------------; ADD2 = 0
                                ;13360          R[TEMP4]_0,                     ;
                                ;13361          CLEAR OPZERO(FLAG3),            ; BE SURE FLAG IS ZERO
U 05CA, 081C,05B7,0031,0047,006F,0 ;13362       NEXT/FP.ZEROTEMP1               ; ZERO OUT OTHER HALF
                                ;13363                                          ; DOES A RETURN +1 TO CALLING RTN
                                ;13364
                                ;13365          ;11-----------------------------; ADD2 > 0
                                ;13366          R[TEMP4]_M[MDR].RR.16,          ; ROTATE BITS 32-63
U 05CB, 040D,23B7,0011,0047,0063,F ;13367       CLEAR ADD2(FLAG1)               ; BE SURE FLAG IS CLEAR
                                ;13368
                                ;13369 FP.ADDDOP2.20:
                                ;13370          ;-------------------------------;
U 063F, 0086,1677,0031,0047,0064,7 ;13371       M[TEMP1]_UNPACK(MB R[TEMP4])    ; UNPACK BITS 0-31
                                ;13372
                                ;13373          ;-------------------------------;
                                ;13374          R[TEMP4]_RB.ASL.7,              ; GETS BITS 32-63 OF FRACTION
U 0647, 0C84,0577,00B1,0047,0000,1 ;13375       RETURN [+1]                     ; RETURN TO CONTINUE
```

```
;13376 .TOC "  Floating point and CRC          : FP.ADDD.20 FP.ACBD.100 "
;13377
;13378 ;*****************************************************************************
;13379 ;
;13380 ;               FP.ADDD.20
;13381 ;
;13382 ;                       INTERFACE FOR ADDD2/3,SUBD2/3,ACBD
;13383 ;
;13384 ;               FP.ACBD.100
;13385 ;
;13386 ;                       Input for FP.ACBD.100
;13387 ;                               Temp1           addend.rd (bits 0-31)
;13388 ;                               MDR             (bits 32-63)
;13389 ;
;13390 ;                       Subroutines:
;13391 ;                               FP.ADDDOP1      parse addend.rd
;13392 ;                               OS.DMOD         get index.md
;13393 ;
;13394 ;                       **This entry through IRDROM saves 1 word and cycle
;13395 ;                       and allows the return from OS.DMOD to be +1 and
;13396 ;                       fall into the common code for the add double interface
;13397 ;
;13398 ;
;13399 ;                       Input:
;13400 ;                               Temp1           "src2" operand (bits 0-31)
;13401 ;                               MDR             (bits 32-63)
;13402 ;
;13403 ;                          for src1
;13404 ;                               Temp2,Temp3             fraction
;13405 ;                               Temp5                   exponent
;13406 ;                               Flag0                   sign of operand
;13407 ;                                  (ADD1(FLAG0))
;13408 ;
;13409 ;               Resources:
;13410 ;                  same as above for src1
;13411 ;
;13412 ;                               Temp1           fraction src2 (bits 0-31)
;13413 ;                               Temp4           (bits 32-63)
;13414 ;                               Temp7,Q         exponent
;13415 ;                               Flag1           sign of operand
;13416 ;                                  (ADD2(FLAG1))
;13417 ;
;13418 ;                               * Temp7 also contains the difference between
;13419 ;                               the exponents during the operation
;13420 ;                               Platch          exponent difference
;13421 ;
;13422 ;                               *Flag1 (WRITE(FLAG1)) also  is cleared after
;13423 ;                               the add operation for later writing
;13424 ;
;13425 ;               Subroutines:
;13426 ;                               FP.ADDDOP2              parse src2
;13427 ;                               FP.ADDDRTN              to perform add function
```

```
                                ;13428 ;                       Exits:
                                ;13429 ;                               FP.NORM.DBL             normalize,round,pack
                                ;13430 ;                                                       answer
                                ;13431 ;                               return from this routine is to the main
                                ;13432 ;                               calling routine
                                ;13433 ;                               except if overflow or underflow and FU bit set
                                ;13434 ;
                                ;13435 ;                       Output:
                                ;13436 ;                               to FP.NORM.DBL
                                ;13437 ;
                                ;13438 ;                               Temp2,Temp3             fraction answer
                                ;13439 ;                               Temp5                   exponent
                                ;13440 ;                               D                       sign of answer
                                ;13441 ;
                                ;13442 ;*******************************************************************************
                                ;13443
                                ;13444 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;13445 =000
                                ;13446 FP.ACBD.100:
                                ;13447         ;000-----------------------------; IRDROM ENTRY
                                ;13448         R[TEMP5]_EXP(M[TEMP1]),         ; WITH ADD.RD IN TEMP1,MDR
                                ;13449         FRO.FLTZ?,                      ; TEST VALIDITY OF OPERAND
U 00A0, 0C84,1437,0AB1,4047,0C5C,4 323*  ;13450         PUSH,NEXT/FP.ADDDOP1            ;
                                ;13451
                                ;13452         ;001-----------------------------; Q BACK TO PROPER TEMP
                                ;13453         M[TEMP2]_Q,                     ;
                                ;13454         PUSH,                           ; GO GET INDEX
                                ;13455         LOD INC BRA?,                   ;
U 00A1, 0086,203A,41BD,8047,0419,0       ;13456         NEXT/OS.DMOD                    ; RETURN IS +1
                                ;13457
                                ;13458 FP.ADDD.20:
                                ;13459         ;010-----------------------------; ENTRY POINT FOR PARSE OF ADD2
                                ;13460         R[TEMP7]_Q_EXP(M[TEMP1]),       ; SAVE EXPONENT IN Q ALSO
                                ;13461         FRO.FLTZ?,                      ; TEST VALIDITY OF OPERAND
U 00A2, 0884,1435,1AB1,C047,0C5C,8 389*  ;13462         PUSH,NEXT/FP.ADDDOP2            ;
                                ;13463
                                ;13464 FP.ADDD.25:                              ; NEW LABEL FOR FIX
                                ;13465         ;011-----------------------------;
                                ;13466         R[TEMP7]_PL_M[TEMP5]-RB,        ; DIFFERENCE BETWEEN EXPONENTS
                                ;13467         SIGND CMP?,SIZE[WORD],          ; SAVE ALSO IN PLATCH
                                ;13468         PUSH,                           ; SAVE FOR RETURN FROM HEART OF ADD
U 00A3, 0884,5BC0,0B51,C047,0C52,0 374*  ;13469         NEXT/FP.ADDDRTN                 ; OFF TO THE MAIN PART OF ADD
                                ;13470
                                ;13471         ;100-----------------------------;
                                ;13472         R[TEMP2]_M[MDR],                ; BITS 0-31 IN MDR AT RETURN
U 00A4, 080D,2592,4030,8047,0064,F       ;13473         CLEAR WRITE(FLAG1)              ; BITS 32-63 IN TEMP3
                                ;13474 =
                                ;13475 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;13476 FP.ADDD.30:
                                ;13477         ;-----------------------------;
                                ;13478         M[TEMP5]_MB-ZLIT0[1E],          ; SUB THE FACTOR
                                ;13479         CLEAR MUL1(FLAG2),              ;
U 064F, 0116,5C10,0030,F047,0068,D       ;13480         NEXT/FP.NORM.DBL                ; OFF TO NORMALIZATION
```

```
;13481 .TOC "  Floating point and CRC          : FP.ADDDRTN "
;13482
;13483 ;*****************************************************************************
;13484 ;
;13485 ;                FP.ADDDRTN
;13486 ;
;13487 ;                        HEART OF DOUBLE PRECISION ADD ROUTINE
;13488 ;                        USED BY ADDD2,ADDD3,SUBD2,SUBD3,ACBD,POLYD
;13489 ;
;13490 ;
;13491 ;                        Input:
;13492 ;
;13493 ;                                Platch          difference between exponents
;13494 ;                                Temp7           difference between exponents
;13495 ;                                Temp2           fraction src1 (bits 0-31)
;13496 ;                                Temp3           fraction src1 (bits 32-63)
;13497 ;                                Temp5           exponent src1
;13498 ;                                Flag0           src1 sign
;13499 ;                                   (ADD1(FLAG0))
;13500 ;                                Temp1           fraction src2 (bits 0-31)
;13501 ;                                Temp4           fraction src2 (bits 32-63)
;13502 ;                                Q               exponent
;13503 ;                                Flag1           src2 sign
;13504 ;                                   (ADD2(FLAG1))
;13505 ;
;13506 ;                        Resources:
;13507 ;                                MDR             bits 0-31 of working temp
;13508 ;                                Temp0           bits 32-63 of working temp
;13509 ;
;13510 ;                                Flag 4          different sign flag
;13511 ;                                   (SAMESIGN(FLAG4))
;13512 ;
;13513 ;                Subroutines:
;13514 ;                        FP.ADDD.SIGNOP1
;13515 ;                        FP.ADDD.SIGNOP2
;13516 ;
;13517 ;
;13518 ;
;13519 ;                        Output:
;13520 ;                        Temp2,Temp3     fraction
;13521 ;                        Temp5           exponent
;13522 ;                        D               sign of answer
;13523 ;
;13524 ;
;13525 ;                Operation:
;13526 ;                        Enters on a three branch testing exponent
;13527 ;                                src1>src2
;13528 ;                                src1=src2
;13529 ;                                src1<src2
;13530 ;
;13531 ;                        The "smaller" value is placed in the working
;13532 ;                        temps (MDR, Temp0) and the appropriate routine
;13533 ;                        entered to set the sign of the answer
;13534 ;                        and to complement the value if required
```

```
                                        ;13535 ;  In the case of equal exponents src2 is selected
                                        ;13536 ;  as the smaller value .***Note only with
                                        ;13537 ;  like signs in the case of equal exponents
                                        ;13538 ;  can the sign of the answer be predetermined
                                        ;13539 ;  Therefore, a flag(SAMESIGN(FLAG4)) is used
                                        ;13540 ;  to differentiate between the signs being same or
                                        ;13541 ;  different
                                        ;13542 ;  For unlike exponents
                                        ;13543 ;  a test is made to see if the difference
                                        ;13544 ;  is < 64. If not then the unpacked fraction and
                                        ;13545 ;  exponent of the larger value is returned
                                        ;13546 ;  to continue through the normalization
                                        ;13547 ;  routine to strip off the hidden bits and pack
                                        ;13548 ;  If the difference is <=63 the smaller
                                        ;13549 ;  value is aligned and bits 32-63 of the fractions
                                        ;13550 ;  are added and the carry if present propagated
                                        ;13551 ;  in the addition of bits 0-31
                                        ;13552 ;  Bits 32-63 of the answer are copied into Temp3
                                        ;13553 ;  and a return effected with bits 0-31 in MDR
                                        ;13554 ;
                                        ;13555 ;  For like exponents
                                        ;13556 ;  The sign and complement operations are
                                        ;13557 ;  performed and the fraction parts added
                                        ;13558 ;  if the operands had unlike signs
                                        ;13559 ;  bit 31 is checked if set the value is
                                        ;13560 ;  complemented and the sign in D inverted
                                        ;13561 ;  Temp3 contains bits 32-63 and MDR has bits 0-31
                                        ;13562 ;  (complemented if necessary) upon return
                                        ;13563 ;
                                        ;13564 ;*******************************************************************************
                                        ;13565
                                        ;13566 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                        ;13567 =000
                                        ;13568 FX.ADDDRTN:                          ; USED BY G FLOAT
                                        ;13569 FP.ADDDRTN:                          ;
                                        ;13570         ;000-------------------------; SRC1 > SRC2  (EXPONENTS)
                                        ;13571         MDR_R[TEMP1],                ; SMALLER VALUE IN WORKING TEMPS
                                        ;13572         PUSH,                        ; MDR-BITS0-31 TEMP0 32-63
U 0520, 0080,05BE,4030,4467,0460,7      ;13573         NEXT/FP.ADD.SIGNOP1          ; TO SIGN TESTING RET IS +3
                                        ;13574
                                        ;13575         ;001-------------------------; SRC1 = SRC2  (EXPONENTS)
                                        ;13576         MDR_R[TEMP1],                ; NO ALIGN NEEDED MAY NEED INVERSION
                                        ;13577         PUSH,                        ; RETURN IS +3
U 0521, 0080,05BE,4030,4467,0460,7      ;13578         NEXT/FP.ADD.SIGNOP1          ;
                                        ;13579
                                        ;13580         ;010-------------------------; SRC1 < SRC2  (EXPONENTS)
                                        ;13581         MDR_R[TEMP2],                ; SET SMALLER VALUE IN WORKING TEMPS
U 0522, 0080,05BE,4030,8467,005C,C      ;13582         NEXT/FP.ADDDRTN.120          ; (T2,T3,T5E)
                                        ;13583
                                        ;13584         ;011-------------------------; RET + 3 FROM SRC1>SRC2
                                        ;13585         M[TEMP7]_MB-ZLIT0[32.],      ; IS EXP DIFF TOO BIG
                                        ;13586         WB<31-30>?,                  ; TEMP7 HAS DIFF (Q HAS SRC2 EXP)
                                        ;13587         PUSH,                        ; RET +3 IF IN RANGE
U 0523, 0586,7C10,06F1,0047,0C5D,9 377* ;13588         NEXT/FP.ADDDRTN.RNGCHK       ; RET +2 IF OUT OF RANGE ( ANS IS SRC1)
```

```
                                    ;13589          ;100------------------------------; RET +3 FROM SRC1=SRC2
                                    ;13590          R[TEMP0]_M[TEMP3]+RB,            ; FIRST PART OF ADD OPERATION
                                    ;13591          SAMESIGN(FLAG4)?,                ; CHECK IF EQU EXP DIFF SIGNS
U 0524, 0084,3001,0470,0047,0069,C  ;13592          NEXT/FP.ADDDRTN.200              ;
                                    ;13593
                                    ;13594          ;101------------------------------; RET +2 FROM RANGE CHECK
                                    ;13595          MDR_R[TEMP2],                    ; THO ANS IS SRC1 ( >63 DIFF)
U 0525, 0C80,05BE,40B0,8467,0000,1  ;13596          RETURN [+1]                      ; COST LESS TO DO NORMALIZE TO GET
                                    ;13597                                           ; ANS PACKED IE HAVE TO STRIP OFF
                                    ;13598                                           ; HIDDEN BIT AND THEN PACK SO DO
                                    ;13599                                           ; DUMMY MOVE HERE FOR COMMON RETURN
                                    ;13600
                                    ;13601          ;110------------------------------; RET +3 FROM RANGE CHECK
                                    ;13602          R[TEMP0]_M[TEMP3]+RB,            ; ADD FOR SRC1 >SRC2
U 0526, 0884,3001,0030,0047,0069,D  ;13603          NEXT/FP.ADDDRTN.250              ; DO SECOND PART
                                    ;13604 =
                                    ;13605
                                    ;13606 =00
                                    ;13607 FP.ADDDRTN.120:
                                    ;13608          ;00------------------------------; SRC1 < SRC2 (EXPONENTS)
                                    ;13609          M[TEMP0]_R[TEMP3],               ; GO CHECK SIGN
                                    ;13610          ADD2(FLAG1) ADD1(FLAG0)?,        ;
U 05CC, 0C86,05BE,4530,C047,045B,4  ;13611          PUSH,NEXT/FP.ADD.SIGNOP2         ; RETURN +3
                                    ;13612
                                    ;13613 =11
                                    ;13614          ;11------------------------------; RET +3 INVERTED IF NECESSARY
U 05CF, 0884,5BCB,0031,C047,0065,4  ;13615          R[TEMP7]_PL_Q-M[TEMP5]           ; COMPUTE DIFF  EXP  IN TEMP7 AND PL
                                    ;13616
                                    ;13617          ;--------------------------------;
U 0654, 0C86,503A,403D,8047,005D,0  ;13618          M[TEMP5]_Q                       ; PUT SRC2 EXP IN ANS EXP TEMP
                                    ;13619
                                    ;13620 =00
                                    ;13621          ;00------------------------------; CHECK EXP TOO BIG
                                    ;13622          M[TEMP7]_MB-ZLIT0[32.],          ; RET +3 IF IN RANGE
                                    ;13623          WB<31-30>?,                      ; RET +2 IF OUT OF RANGE
                                    ;13624          PUSH,                            ;
U 05D0, 0586,7C10,06F1,0047,0C5D,9 377* ;13625      NEXT/FP.ADDDRTN.RNGCHK           ;
                                    ;13626
                                    ;13627 =10
                                    ;13628          ;10------------------------------; RET +2  OUT OF RANGE
                                    ;13629          MDR_R[TEMP1],                    ; ANS IS SRC2 BUT THE COST TOO HIGH
U 05D2, 0080,05BE,4030,4467,0065,6  ;13630          NEXT/FP.ADDDRTN.125              ; DO STRIPPING AND PACKING
                                    ;13631                                           ; IN COMMON FLOWS MOVING VALUES TO
                                    ;13632                                           ; CORRECT TEMPS
                                    ;13633
                                    ;13634          ;11------------------------------; RET +3 IN RANGE
                                    ;13635          R[TEMP0]_M[TEMP4]+RB,            ; FIRST PART OF ADD
U 05D3, 0C84,4001,0030,0047,0065,8  ;13636          NEXT/FP.ADDDRTN.130              ;
```

```
                                ;13637 FX.T3GETST4:
                                ;13638 FP.ADDDRTN.125:
                                ;13639          ;--------------------------------;
                                ;13640          R[TEMP3]_M[TEMP4],              ; MOVE BITS 32-63
U 0656, 0C84,4592,40B0,C047,0000,1 ;13641       RETURN [+1]                     ; RETURN  +1 TO COMMON FLOWS
                                ;13642
                                ;13643 FP.ADDDRTN.130:
                                ;13644          ;--------------------------------;
                                ;13645          MDR_M[MDR]+R[TEMP1]+ALKC,       ; SECOND PART OF ADD
U 0658, 0C81,2041,0030,4467,005D,5 ;13646       NEXT/FP.ADDDRTN.300             ; FINISH UP PROCESSING
                                ;13647
                                ;13648 =0
                                ;13649 FP.ADDDRTN.200:
                                ;13650          ;0-------------------------------; SECOND PART OF ADD
                                ;13651          MDR_M[MDR]+R[TEMP2]+ALKC,       ; FOR EQU EXP AND DIFF SIGNS
                                ;13652          WB<31-30>?,                     ; TEST SIGN OF SUM
U 069C, 0481,2041,06F0,8467,085D,5 338* ;13653  NEXT/FP.ADDDRTN.300             ;
                                ;13654
                                ;13655 FP.ADDDRTN.250:
                                ;13656          ;1-------------------------------; SECOND PART OF ADD FOR
U 069D, 0C81,2041,0030,8467,005D,5 ;13657       MDR_M[MDR]+R[TEMP2]+ALKC        ; EQU EXP SAME SIGN AND
                                ;13658                                          ; AND SR1>SRC2
                                ;13659
                                ;13660 =01
                                ;13661 FP.ADDDRTN.300:
                                ;13662          ;01------------------------------;
                                ;13663          R[TEMP3]_M[TEMP0],              ; COPY BITS 32-63
                                ;13664          CLEAR SAMESIGN(FLAG4),          ;
U 05D5, 0824,0592,40B0,C047,0000,1 ;13665       RETURN [+1]                     ;
                                ;13666
                                ;13667 =11
                                ;13668 FP.ADDDRTN.320:
                                ;13669          ;11------------------------------;
U 05D7, 0D80,0D73,6030,4047,0065,A ;13670       D_D.XOR.ZLIT12[8]               ; FLIP THE SIGN BIT AND RETURN
                                ;13671
                                ;13672          ;--------------------------------;
                                ;13673          R[TEMP3].SIZ_-M[TEMP0],         ; COMPLEMENT VALUE
U 065A, 0082,0593,0020,C047,0066,0 ;13674       SIZE[LONG]                      ; BITS 32-63
                                ;13675
                                ;13676          ;--------------------------------;
                                ;13677          MDR_R[ZERO]-M[MDR]-ALKC,        ; COMPL BITS 0-31
U 0660, 0C81,2043,00AD,8467,0000,1 ;13678       SIZE[LONG],RETURN [+1]          ;
```

```
                                        ;13679 =01
                                        ;13680 FP.ADDDRTN.RNGCHK:
                                        ;13681          ;01-------------------------------; DIFF >=32
                                        ;13682          WB_ZLIT0[31.]-M[TEMP7],         ; DIFF > 63?
                                        ;13683          WB<31-30>?,                     ;
U 05D9, 0980,7C13,06F0,F847,085D,D 377* ;13684          NEXT/FP.ADDDRTN.620             ;
                                        ;13685
                                        ;13686
                                        ;13687          ;11-------------------------------; DIFF 1-31
U 05DB, 0C85,2137,0030,0047,0066,2      ;13688          R[TEMP0]_(M[MDR] RB).RR.P       ; ALIGN SMALLER VALUE BITS 32-63
                                        ;13689                                          ; PLATCH HAS BEEN PREVIOUSLY SET
                                        ;13690 =
                                        ;13691          ;-------------------------------;
                                        ;13692          MDR_M[MDR].ASR.P,               ; BITS 0-31
U 0662, 0081,20F7,00B0,0467,0000,3      ;13693          RETURN [+3]                     ;
                                        ;13694
                                        ;13695 =01
                                        ;13696 FP.ADDDRTN.620:
                                        ;13697          ;01-------------------------------; DIFF  (32-63)
                                        ;13698          PL_M[TEMP7],                    ; LOAD PLATCH WITH THE VALUE
U 05DD, 0880,7BC2,403D,8047,0066,7      ;13699          NEXT/FP.ADDDRTN.625             ; PREVIOUSLY CALCULATED RANGE0-31
                                        ;13700
                                        ;13701          ;11-------------------------------; DIFF  >63
U 05DF, 0980,0EF6,4030,F847,0066,7      ;13702          PL_[1F]                         ; SET TO 31 NEED FOR POLYD ACCURACY
                                        ;13703
                                        ;13704 FP.ADDDRTN.625:
                                        ;13705          ;-------------------------------; THIS FUNCTION SHIFTS IN BIT 31 OF MDR
                                        ;13706          R[TEMP0]_M[MDR].ASR.P,           ; GET THE FRACTION BITS
U 0667, 0085,20F7,06F0,0047,005E,1      ;13707          WB<31-30>?                      ; WHAT ARE BIT SETTINGS IN 0-31
                                        ;13708
                                        ;13709 =01
                                        ;13710          ;01-------------------------------; BITS 0-31 POSITIVE
                                        ;13711          MDR_0,                          ;
U 05E1, 0880,0036,40B0,04E7,0000,3      ;13712          RETURN [+3]                     ;
                                        ;13713
                                        ;13714          ;11-------------------------------; BITS 0-31 TO BE NEGATIVE
                                        ;13715          MDR_-1,                         ;
U 05E3, 0080,0E77,00B0,0467,0000,3      ;13716          RETURN [+3]                     ;
```

```
;13717 .TOC "  Floating point and CRC          : FP.MULD2 FP.MULD3 "
;13718
;13719 ;*****************************************************************************
;13720 ;
;13721 ;       64      MULD2   mulr.rd,prod.md
;13722 ;
;13723 ;       65      MULD3   mulr.rd,muld.rd,prod.wd
;13724 ;
;13725 ;               Input:
;13726 ;                       Temp1                   mulr.rd (bits 0-31)
;13727 ;                       MDR                     (bits 32-63)
;13728 ;
;13729 ;               Resources:
;13730 ;                       for mulr
;13731 ;                       Temp2,Q                 fraction (bits 0-31)
;13732 ;                       Temp3                   fraction (bits 32-63)
;13733 ;                       Temp5                   exponent
;13734 ;                       Flag2                   sign of operand
;13735 ;                          (MUL1(FLAG2))
;13736 ;
;13737 ;               Subroutines:
;13738 ;                       FP.MULDOP1              parse mulr
;13739 ;                       *OS.DMOD                get muld for MULD2
;13740 ;                       *OS.DRED                get muld for MULD3
;13741 ;                       OS.WRT1                 get product for MULD3
;13742 ;                                       * return is through IRDROM to
;13743 ;                                       FP.MULD.20 to continue operation
;13744 ;
;13745 ;                       Exit:
;13746 ;                               FP.WRITE.SEC            write bits 32-63 of answer
;13747 ;
;13748 ;                       Operation:
;13749 ;                               These are the driving routines for the double precision
;13750 ;                               MULD2 and MULD3 instructions
;13751 ;
;13752 ;       **********FPA ENTRY AND INTERFACE**********
;13753 ;       MULD2   (OS.RED) - FI.ADDD2.REG/FI.ADDD2.MEM
;13754 ;       MULD3   (OS.RED) - FI.ADDD3.REG/FI.ADDD3.MEM
;13755 ;
;13756 ;*****************************************************************************
```

```
                                ;13757 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;13758 =00
                                ;13759 FP.MULD2:                                ; **********
                                ;13760          ;00-----------------------------;
                                ;13761          R[TEMP5]_EXP(M[TEMP1]),         ; MULTIPLIER PARSE
                                ;13762          FRO.FLTZ?,                      ; TEST VALIDITY OF OPERAND
U 00B8, 0484,1437,0AB1,4047,0C5E,4 323*         ;13763          PUSH,NEXT/FP.MULDOP1            ; AND UNPACK THE FRACTION
                                ;13764
                                ;13765          ;01-----------------------------;
                                ;13766          M[TEMP2]_Q,                     ; BITS 0-31 TO TEMP
                                ;13767          LOD INC BRA?,                   ; RET THRU IRDROM TO FP.MULD.20
U 00B9, 0086,203A,41BD,8047,0419,0              ;13768          PUSH,NEXT/OS.DMOD               ; FINAL RETURN AFTER ANSWER
                                ;13769
                                ;13770          ;10-----------------------------;
                                ;13771          R[DST.R]_M[TEMP2],              ; WRITE FIRST PART
                                ;13772          WRITE NOTREG,SIZE[IDEP],        ; NO INTERRUPTS ALLOWED
                                ;13773          CCOP1,SET MM.NOINT,             ; SET CONDITION CODES
U 00BA, 0064,2592,403C,4DDA,0056,F              ;13774          NEXT/FP.WRITE.SEC               ;
                                ;13775 =
                                ;13776 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;13777 =00
                                ;13778 FP.MULD3:                                ; **********
                                ;13779          ;00-----------------------------;
                                ;13780          R[TEMP5]_EXP(M[TEMP1]),         ; MULTIPLIER PARSE
                                ;13781          FRO.FLTZ?,                      ; TEST VALIDITY OF OPERAND
U 00C8, 0484,1437,0AB1,4047,0C5E,4 323*         ;13782          PUSH,NEXT/FP.MULDOP1            ; AND UNPACK THE FRACTION
                                ;13783
                                ;13784          ;01-----------------------------;
                                ;13785          M[TEMP2]_Q,                     ; BITS 0-31 TO TEMP
                                ;13786          PUSH,                           ; GET NEXT OPERAND
                                ;13787          LOD INC BRA?,                   ; RET THRU IRDROM TO FP.MULD.20
U 00C9, 0886,203A,41BD,8047,0418,0              ;13788          NEXT/OS.DRED                    ;
                                ;13789
                                ;13790          ;10-----------------------------;
                                ;13791          PUSH,                           ; RET TO GET THE DESTINATION
                                ;13792          LOD INC BRA?,                   ;
U 00CA, 0C80,0036,41B0,0047,0414,0              ;13793          NEXT/OS.WRT1                    ;
                                ;13794
                                ;13795          ;11-----------------------------;
                                ;13796          R[DST.R]_M[TEMP2],              ; WRITE FIRST PART
                                ;13797          WRITE NOTREG, SIZE[IDEP],       ;
                                ;13798          SET MM.NOINT,                   ; NO INTERRUPTS ALLOWED
                                ;13799          CCOP1,                          ; SET CONDITION CODES
U 00CB, 0064,2592,403C,4DDA,0056,F              ;13800          NEXT/FP.WRITE.SEC               ;
                                ;13801 =
                                ;13802 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
```

```
;13803 .TOC "  Floating point and CRC          : FP.MULD.20 "
;13804
;13805 ;*****************************************************************************
;13806 ;
;13807 ;                          FP.MULD.20
;13808 ;
;13809 ;       INTERFACE FOR THE DOUBLE PRECISION MULTIPLY INSTRUCTIONS
;13810 ;
;13811 ;               Input:
;13812 ;                       Temp1                   multiplicand bits 0-31
;13813 ;                       MDR                     bits 32-63
;13814 ;                       for multiplier
;13815 ;                       Temp2,Temp3             fraction
;13816 ;                       Temp5                   exponent
;13817 ;                       Flag2                   sign of multiplier
;13818 ;                          (MUL1(FLAG2))
;13819 ;
;13820 ;               Resources:
;13821 ;                       for multiplier
;13822 ;                       same as for input
;13823 ;                       for multiplicand
;13824 ;                       Temp1                   fraction (bits 0-31)
;13825 ;                       Temp6                   (bits 32-63)
;13826 ;                       RTemp8          exponent
;13827 ;                       Flag3                   sign of multiplicand
;13828 ;                          (MUL2(FLAG3))
;13829 ;                       D,Q                     for multiply operation
;13830 ;                       Stepcounter             for loop control
;13831 ;
;13832 ;               Subroutines:
;13833 ;                       FP.MULDOP2              parse multiplicand
;13834 ;                       FP.SC17                 set stepcounter to 17
;13835 ;                       FP.PREOPS.SUB           prepare operands and
;13836 ;                                               call multiply subroutine
;13837 ;                                               (FP.MULD.SUB)
;13838 ;               Exit:
;13839 ;                       FP.NORM.MUDI            to normalize,round,pack
;13840 ;                                               and return to calling routine
;13841 ;               Operation:
;13842 ;                       The multiplicand is validated and parsed by FPMULDOP2
;13843 ;                       If either operand is 0, product is 0 return to caller
;13844 ;                       Otherwise, the exponents are added interrupts & timer
;13845 ;                       are checked stepcounter = 17. "Fudge" factor is
;13846 ;                       subtracted from sum of exponents (80 for bias
;13847 ;                       and +19-hex for the binary point and platch)
;13848 ;                       FP.PREOPS.SUB is called to shift operands left 1
;13849 ;                       remove the overflow bit and call FP.MULD.SUB
;13850 ;                       to perform the cross products multiplication
;13851 ;                       bits 0-31 of the product are derived by D+ALKC and
;13852 ;                       the carry in Temp2 from the previous addition
;13853 ;                       The product sign is derived by calling FP.NORM.MUDI
;13854 ;                       with the XOR of the operands signs
;13855 ;
;13856 ;*****************************************************************************
```

```
                                        ;13857 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;13858 =000
                                        ;13859 FP.MULD.20:
                                        ;13860          ;000-------------------------------;
                                        ;13861          R[TEMP8]_EXP(M[TEMP1]),           ; RET WITH SECOND OPERAND
                                        ;13862          FRO.FLTZ?,                        ; TEST VALIDITY AND PARSE
U 00C0, 0084,1437,0AB2,0047,0C5E,8 323* ;13863          PUSH,NEXT/FP.MULDOP2              ;
                                        ;13864
                                        ;13865          ;001-------------------------------; ANS IS ZERO
                                        ;13866          R[TEMP2]_FLAGS_0,                 ; RETURN TO FIND OUT WHICH
U 00C1, 0484,058E,00B0,8307,0000,1      ;13867          RETURN [+1]                       ; INSTRUCTION
                                        ;13868
                                        ;13869          ;010-------------------------------; RET +2
                                        ;13870          M[TEMP5]_MB+R[TEMP8],             ; MULTIPLICAND BITS 0-31 TEMP1
                                        ;13871          SIZE[WORD],                       ; BITS 32-63 TEMP6
U 00C2, 0C86,5001,0012,0047,046A,6      ;13872          PUSH,NEXT/FP.SC17                 ; EXP RTEMP8
                                        ;13873
                                        ;13874          ;011-------------------------------;
                                        ;13875          M[TEMP5]_MB-ZLIT0[9F],            ; SUB FACTORS
                                        ;13876          SIZE[WORD],                       ; 19 FOR PLATCH AND BP
                                        ;13877          PUSH,                             ; 80 FOR EXTRA BIAS
U 00C3, 0586,5C10,0014,F847,046A,2      ;13878          NEXT/FP.PREOPS.SUB                ; SHIFT OPS AND MULTIPLY
                                        ;13879
                                        ;13880          ;100-------------------------------; VALUE CAN BE 01 OR 10
                                        ;13881          M[TEMP2]_D+R[TEMP0]+ALKC,         ; BITS 0-31 D+CARRY
                                        ;13882                                            ; AND CARRY IN TEMP0 FROM PREVIOUS ADD
                                        ;13883                                            ; BITS 32-63 TEMP3
                                        ;13884          MUL1(FLAG2) XOR MUL2(FLAG3)?,     ; FLAG1=0,FLAG2XOR3 IS PRODUCT SIGN
U 00C4, 0486,2061,0570,0047,0050,A      ;13885          NEXT/FP.NORM.MUDI                 ; FINAL RETURN INITIAL CALLER
                                        ;13886                                            ; EXCEPT IF FLOATING OVERFLOW
                                        ;13887                                            ; OR UNDERFLOW AND FU BIT SET
                                        ;13888 =
                                        ;13889 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
```

```
;13890 .TOC "  Floating point and CRC          : FP.MULDOP1 FP.MULDOP2 "
;13891
;13892 ;*****************************************************************************
;13893 ;
;13894 ;       DOUBLE PRECISION MULD AND DIVD OPERAND PARSERS
;13895 ;
;13896 ;               FP.MULDOP1
;13897 ;
;13898 ;                       Input:
;13899 ;                               Temp1           bits 0-31 ("src1")
;13900 ;                               MDR             bits 32-63
;13901 ;                               Temp5           exponent
;13902 ;                               Flag2           clear
;13903 ;                                  (MUL1(FLAG2))
;13904 ;                               Flag3           clear
;13905 ;                                  (MOPZERO(FLAG3))
;13906 ;
;13907 ;                       Resources:
;13908 ;                               Q               fraction bits 0-31
;13909 ;                               Temp3           fraction bits 32-63
;13910 ;                               Flag2           sign of operand
;13911 ;                                  (MUL1(FLAG2))
;13912 ;                               Flag3           set if operand 0
;13913 ;                                  (MOPZERO(FLAG3))
;13914 ;                       Exit:
;13915 ;                               Return +1       if operand zero
;13916 ;                               FP.ADDDOP1.20   common unpack  then return +1
;13917 ;                               IE.OPER.FAULT   reserved operand
;13918 ;
;13919 ;               FP.MULDOP2
;13920 ;
;13921 ;                       Input:
;13922 ;                               Temp1           bits 0-31 ("src2")
;13923 ;                               MDR             bits 32-63
;13924 ;                               Temp2,Temp3     fraction "src1"
;13925 ;                               Temp5           exponent
;13926 ;                               Flag2           sign of "src1"
;13927 ;                                  (MUL1(FLAG2))
;13928 ;                               Flag1           may be set or reset
;13929 ;                                  (MOPZERO(FLAG1))
;13930 ;                               Flag3           clear
;13931 ;                                  (MUL2(FLAG3))
;13932 ;
;13933 ;                       Resources:
;13934 ;                               Temp1           fraction bits 0-31 ("src2")
;13935 ;                               Temp6           fraction bits 32-63
;13936 ;                               RTemp8          exponent "src2"
;13937 ;                       Exit:
;13938 ;                               Return +2       if either operand 0
;13939 ;                               Return +1       if neither operand 0
;13940 ;                               IE.OPER.FAULT   reserved operand
;13941 ;
;13942 ;*****************************************************************************
```

```
                                ;3943 =00
                                ;3944 FP.MULDOP1:
                                ;13945          ;00-------------------------------; RESERVED OPERAND
U 05E4, 0C80,0036,4030,0047,00FF,8      ;13946          NEXT/IE.OPER.FAULT              ;
                                ;13947
                                ;13948          ;01-------------------------------; SRC1 < 0
                                ;13949          R[TEMP3]_M[MDR].RR.16,          ; ROTATE BITS 32-63
                                ;13950          SET MUL1?(FLAG2),               ; SIGNIFY NEGATIVITY
U 05E5, 0055,23B7,0010,C047,005D,C      ;13951          NEXT/FP.ADDDOP1.20              ; COMMON FLOWS WITH OTHER DOUBLES
                                ;13952
                                ;13953          ;10-------------------------------; SRC1 = 0
                                ;13954          R[TEMP3]_Q_0,MDR_0,             ; SET OPERANDS TO ZERO
                                ;13955          SET MOPZERO(FLAG1),             ; SET FLAG TO SIGNIFY THIS
U 05E6, 004C,05B7,1030,C4E7,006F,0      ;13956          NEXT/FX.ZEROTEMP1               ; ZERO OUT BITS 0-31 PACKED FORM
                                ;13957
                                ;13958          ;11-------------------------------; SRC1 > 0
                                ;13959          R[TEMP3]_M[MDR].RR.16,          ; ROTATE BITS 32-63
U 05E7, 0885,23B7,0010,C047,005D,C      ;13960          NEXT/FP.ADDDOP1.20              ; CONTINUE COMMON FLOWS FOR PARSING
                                ;13961
                                ;13962 =00
                                ;13963 FP.MULDOP2:
                                ;13964          ;00-------------------------------; RESERVED OPERAND
U 05E8, 0C80,0036,4030,0047,00FF,8      ;13965          NEXT/IE.OPER.FAULT              ;
                                ;13966
                                ;13967          ;01-------------------------------; SRC2 < 0
                                ;13968          R[TEMP6]_M[MDR].RR.16,          ; ROTATE BITS 32-63
                                ;13969          SET MUL2?(FLAG3),               ; SIGNIFY NEGATIVITY
U 05E9, 085D,23B7,0011,8047,0066,B      ;13970          NEXT/FP.MULDOP?.20              ; CONTINUE UNPACKING
                                ;13971
                                ;13972          ;10-------------------------------; SRC2 = 0
                                ;13973          R[TEMP3]_Q_0,MDR_0,             ; ZERO OUT ANS TEMPS
U 05EA, 0484,05B7,10B0,C4E7,0000,1      ;13974          RETURN [+1]                     ; ANS IS ZERO SEE NOTE BELOW
                                ;13975
                                ;13976          ;11-------------------------------; SRC2 > 0
U 05FB, 0885,23B7,0011,8047,0066,B      ;13977          R[TEMP6]_M[MDR].RR.16           ; ROTATE BITS 32-63
                                ;13978
                                ;13979 FP.MULDOP2.20:
                                ;13980          ;-------------------------------;
                                ;13981          M[TEMP1]_UNPACK(MB R[TEMP6]),   ; UNPACK
U 066B, 0C8o,1677,05F1,8047,005E,C      ;13982          MOPZERO(FLAG1)?                 ; WAS SRC1 0
                                ;13983
                                ;13984 =0*
                                ;13985          ;0*-------------------------------; MOPZERO(FLAG1) NOT SET
                                ;13986          R[TEMP6]_RB.ASL.7,              ; GET BITS 32-63
U 05EC, 0884,0577,00B1,8047,0000,2      ;13987          RETURN [+2]                     ;
                                ;13988
                                ;13989          ;1*-------------------------------; MOPZERO(FLAG1) SET
                                ;13990          R[TEMP3]_Q_0,                   ; ZERO OUT ANS TEMPS
                                ;13991          MDR_0,                          ; TEST OUTSIDE FOR THIS
U 05EF, 0484,05B7,10B0,C4E7,0000,1      ;13992          RETURN [+1]                     ;
```

```
                                ;13993 .TOC "  Floating point and CRC          : FP.PREOPS.SUB "
                                ;13994
                                ;13995 ;*****************************************************************************
                                ;13996 ;
                                ;13997 ;       PREPARATION OF OPERANDS SUBROUTINES
                                ;13998 ;
                                ;13999 ;               FP.PREOPS.SUB - MULD2,MULD3
                                ;14000 ;
                                ;14001 ;               FP.PREOPS.POLYD - POLYD
                                ;14002 ;
                                ;14003 ;               FP.PREOPS.SUBALT - EMODD
                                ;14004 ;
                                ;14005 ;
                                ;14006 ;                       Input and resources:
                                ;14007 ;                               Temp2,Temp3             operand fraction
                                ;14008 ;                               Temp1,Temp6             operand fraction
                                ;14009 ;                               Q
                                ;14010 ;                               Stepcounter
                                ;14011 ;                               MTemp9,MTemp10          operands for POLYD
                                ;14012 ;
                                ;14013 ;                       Operation:
                                ;14014 ;                               For FP.PREOPS.SUB  used by MULD2,MULD3 and
                                ;14015 ;                               POLYD to shift both operands left 1 and
                                ;14016 ;                               thereby remove the overflow bit
                                ;14017 ;                               For FP.PREOPS.POLYD entry is to set flag
                                ;14018 ;                               of argument and to move bits 0-31 from storage
                                ;14019 ;                               to working temp and bits 32-63 also
                                ;14020 ;                               For FP.PREOPS.SUBALT only the operands in
                                ;14021 ;                               Temp1 and Temp6(Q) are shifted
                                ;14022 ;                               For all three entries FP.MULD.SUB is called
                                ;14023 ;                               to perform the double precision multiplication
                                ;14024 ;                               **note interrupts and timer checks are made
                                ;14025 ;                               before calling FP.MULD.SUB
                                ;14026 ;
                                ;14027 ;                               Other subroutines move and save Q and the
                                ;14028 ;                               temps that are used and set the stepcounter
                                ;14029 ;                               and check interrupts and timer
                                ;14030 ;
                                ;14031 ;
                                ;14032 ;*****************************************************************************
                                ;14033
                                ;14034 =01
                                ;14035 FP.PREOPS.POLYD:
                                ;14036         ;01-------------------------------; ARG IS POSITIVE
                                ;14037         CLEAR MUL2(FLAG3),                ; BE SURE FLAG IS CORRECT
                                ;14038         R[TEMP1] M[TEMP9],                ; MOVE OVER FOR POLYD
U 05ED, 001C,9592,4030,4047,0069,E ;14039         NEXT/FP.PREOPS.POLYD10            ;
                                ;14040
                                ;14041         ;11-------------------------------; ARG IS NEGATIVE
                                ;14042         SET MUL2(FLAG3),                  ; BE SURE FLAG IS CORRECT
                                ;14043         R[TEMP1] M[TEMP9],                ; MOVE OVER FOR POLYD
U 05EF, 085C,9592,4030,4047,0069,E ;14044         NEXT/FP.PREOPS.POLYD10            ;
```

```
                                ;14045 =0
                                ;14046 FP.PREOPS.POLYD10:
                                ;14047          ;0-------------------------------;
                                ;14048          Q_M[TEMP3],                     ; PUT BITS 32-63 IN Q
U 069E, 0C80,3592,5030,0047,0467,2 ;14049       PUSH,NEXT/FP.PREOPS.QT2         ;
                                ;14050
                                ;14051          ;1-------------------------------;
                                ;14052          R[TEMP3]_Q Q_M[TEMP10],         ; Q OLD TO TEMP3 AND TEMP10 TO Q
U 069F, 0884,A00C,7030,C047,0066,E ;14053       NEXT/FP.PREOPS.SUBALT           ; CONTINUE ON
                                ;14054
                                ;14055 =0
                                ;14056 FX.PREOPS.POLYGT9:
                                ;14057          ;0-------------------------------;
                                ;14058          Q_M[TEMP3],                     ; PUT BITS 32-63 IN Q
U 06A0, 0C80,3592,5030,0047,0467,2 ;14059       PUSH,NEXT/FP.PREOPS.QT2         ;
                                ;14060
                                ;14061          ;1-------------------------------;
                                ;14062          R[TEMP3]_Q Q_M[TEMP9],          ; Q OLD TO TEMP3 AND TEMP9 TO Q
U 06A1, 0884,900C,7030,C047,0066,E ;14063       NEXT/FP.PREOPS.SUBALT           ; CONTINUE ON
                                ;14064
                                ;14065 =0
                                ;14066 FX.PREOPS.SUB:                           ; USED BY G FLOAT
                                ;14067 FP.PREOPS.SUB:                           ;
                                ;14068          ;0-------------------------------;
                                ;14069          Q_M[TEMP3],                     ; PUT BITS 32-63 IN Q
U 06A2, 0C80,3592,5030,0047,0467,2 ;14070       PUSH,NEXT/FP.PREOPS.QT2         ;
                                ;14071
                                ;14072          ;1-------------------------------;
U 06A3, 0884,600C,7030,C047,0066,E ;14073       R[TEMP3]_Q Q_M[TEMP6]           ; Q OLD TO TEMP3 AND TEMP6 TO Q
```

```
                                ;14074 FP.PREOPS.SUBALT:
                                ;14075          ;--------------------------------;
                                ;14076          M[TEMP1]_(MB Q).SL.1,           ; SHIFT TEMP AND Q (ALU AND Q) LEFT 1
U 066E, 0C86,1205,CB3D,98E7,006A,4 ;14077       IP.TS?                          ; INTERRUPT  OR TIMER
                                ;14078
                                ;14079 =0
                                ;14080 FX.PREOPS.SUBALTG:                       ; USED BY G FLOAT
                                ;14081          ;0--------------------------------;
                                ;14082          M[TEMP6]_Q,                     ; Q BACK TO TEMP6 ALSO
U 06A4, 0486,603A,403D,8047,0052,8 ;14083       NEXT/FP.MULD.SUB                ; NOW MULTIPLY ADN RETURN TO CALLING
                                ;14084
                                ;14085          ;1--------------------------------;
                                ;14086          D_ZLIT0[0],                     ; CLEAR OUT D FOR FPD FAULT
                                ;14087          INTPEND OR TIMER?,              ; DO THE TEST HERE
U 06A5, 0980,0C37,2AF0,0047,04F7,0 ;14088       PUSH,NEXT/IE.SERV.IP.TS2        ; TIME TO TEST FOR INTERRUPTS AND TIMER
                                ;14089
                                ;14090 FX.PREOPS.QT3:                           ; USED BY G FLOAT
                                ;14091 FP.PREOPS.QT3:                           ;
                                ;14092          ;--------------------------------;
U 0670, 0080,05BE,5030,C047,0067,2 ;14093       Q_R[TEMP3]                      ;
                                ;14094
                                ;14095 FP.PREOPS.QT2:
                                ;14096          ;--------------------------------;
                                ;14097          M[TEMP2]_(MB Q).SL.1,           ; SHIFT TEMP AND Q (ALU AND Q) LEFT 1
U 0672, 0C86,2205,C0BD,8047,0000,1 ;14098       RETURN [+1]                     ;
                                ;14099
                                ;14100 FP.QT6.SC17:
                                ;14101          ;--------------------------------;
                                ;14102          Q_R[TEMP6],                     ; LOAD UP Q
U 06E2, 0080,05BE,5031,8047,006A,6 ;14103       NEXT/FP.SC17                    ;
                                ;14104
                                ;14105 FP.QT1.SC17.CHKINT:
                                ;14106          ;--------------------------------;
                                ;14107          Q_M[TEMP1],                     ; Q GETS THE OPERAND
U 06E3, 0480,1592,5B30,18E7,006A,6 ;14108       IP.TS?                          ;
                                ;14109 =0
                                ;14110 FP.SC17:
                                ;14111          ;0--------------------------------;
                                ;14112          STEPC_ZLIT0[17.],               ; SET STEP TO 17 ITERATIONS
U 06A6, 0180,0C37,00B0,8907,0000,1 ;14113       RETURN [+1]                     ;
                                ;14114
                                ;14115          ;1--------------------------------;
                                ;14116          D_ZLIT0[0],                     ; CLEAR OUT D FOR FPD FAULT
                                ;14117          INTPEND OR TIMER?,              ; DO THE TEST HERE
U 06A7, 0980,0C37,2AF0,0047,04F7,0 ;14118       PUSH,NEXT/IE.SERV.IP.TS2        ; TIME TO CHECK FOR INTERRUPT OR TIMER
```

```
;14119 .TOC "  Floating point and CRC          : FP.MULD.SUB "
;14120
;14121 ;*************************************************************************************
;14122 ;
;14123 ;
;14124 ;       FP.MULD.SUB
;14125 ;
;14126 ;               DOUBLE PRECISION MULTIPLY USING CROSS PRODUCTS  MULTIPLICATION
;14127 ;
;14128 ;
;14129 ;
;14130 ;                       A(temp2)                B(temp3)
;14131 ;                       ----------              ----------
;14132 ;       src1            | bits 0-31 |           | bits 32-63 |  exp - temp5
;14133 ;                       ----------              ----------
;14134 ;
;14135 ;
;14136 ;                       C(temp1)                D(temp6)
;14137 ;                       ----------              ----------
;14138 ;       src2            | bits 0-31 |           | bits 32-63 |  exp - Rtemp8
;14139 ;                       ----------              ----------
;14140 ;
;14141 ;
;14142 ;
;14143 ;               INPUT:
;14144 ;                       Rbus                    D,Q
;14145 ;
;14146 ;       BD              Temp3                   Temp6 (Q)
;14147 ;
;14148 ;       BC              Temp3                   Temp1(Q)
;14149 ;
;14150 ;       AD              Temp2                   Temp6 (Q)
;14151 ;
;14152 ;       AC              Temp2                   Temp1 (Q)
;14153 ;
;14154 ;
;14155 ;               OUTPUT:
;14156 ;
;14157 ;       AC              PART1 (D)       PART2 (Q)
;14158 ;
;14159 ;       AD                              PART3 (D)       PART4 (Q)
;14160 ;
;14161 ;       BC                              PART5 (D)       PART6 (Q)
;14162 ;
;14163 ;       BD                                              PART7 (D)   PART8 (Q)
;14164 ;
;14165 ;
;14166 ;               OPERATION:
;14167 ;                       Each multiplication done 2 bits at a time
;14168 ;                       yielding a 64 bit product in D,Q
;14169 ;                       These are identified above as the PART1-8
;14170 ;                       After each multiplication the check is made for
;14171 ;                       interrupt and timer
```

```
                                                ;14172 ;
                                                ;14173 ;          The partial product additions are performed as:
                                                ;14174 ;
                                                ;14175 ;                  1. Discard PART 8
                                                ;14176 ;
                                                ;14177 ;                  2. PART7 -> Temp7
                                                ;14178 ;
                                                ;14179 ;                  3. Temp7 + PART6 -> Temp7
                                                ;14180 ;
                                                ;14181 ;                  4. ALKC + PART5 -> Temp0
                                                ;14182 ;
                                                ;14183 ;                  5. PART4 + Temp7 -> Get ALKC
                                                ;14184 ;
                                                ;14185 ;                  6. PART3 + ALKC + Temp0 -> Temp7
                                                ;14186 ;
                                                ;14187 ;                  7. ALKC  -> Temp0
                                                ;14188 ;
                                                ;14189 ;                  8. PART2 + Temp7 -> Temp3
                                                ;14190 ;
                                                ;14191 ;          ***     9. PART1 (D) +ALKC +Temp0 -> Temp2
                                                ;14192 ;
                                                ;14193 ;                          *** upon return to calling routine
                                                ;14194 ;
                                                ;14195 ;          SUBROUTINES:
                                                ;14196 ;
                                                ;14197 ;                  FP.QT1.SC17.CHKINT
                                                ;14198 ;                  FP.QT6.SC17
                                                ;14199 ;
                                                ;14200 ;
                                                ;14201 ;
                                                ;14202 ;*************************************************************************************
                                                ;14203
                                                ;14204 =000
                                                ;14205 FP.MULD.SUB:
                                                ;14206          ;000-----------------------------;
                                                ;14207          MULFAST+ CAND IN R[TEMP3],      ; STEP=17 Q HAS TEMP6
                                                ;14208          DBZ STEPC?,SIZE[LONG],          ; PART 7 AND 8 IN D AND Q
U 0528, 0480,0027,9320,C047,0052,8              ;14209          NEXT/FP.MULD.SUB                ; (B AND D)
                                                ;14210
                                                ;14211          ;001-----------------------------; ALL DONE 1ST PARTIAL PRODUCT
                                                ;14212          R[TEMP7]_D,                     ; SAVE PART 7 IN TEMP7
                                                ;14213          PUSH,                           ; DISCARD PART 8
U 0529, 0484,05B2,4031,C047,046E,3              ;14214          NEXT/FP.QT1.SC17.CHKINT         ; SET UP FOR SECOND PARTIAL PRODUCT
                                                ;14215                                          ; AND CHECK FOR INTERRRUPTS AND TIMER
                                                ;14216 FP.MULD.SUB.10:
                                                ;14217          ;010-----------------------------;
                                                ;14218          MULFAST+ CAND IN R[TEMP3],      ; STEP=17 Q HAS TEMP1
                                                ;14219          DBZ STEPC?,SIZE[LONG],          ; PART 5 &6 IN D AND Q
U 052A, 0C80,0027,9320,C047,0052,A              ;14220          NEXT/FP.MULD.SUB.10             ; ( B AND C)
                                                ;14221
                                                ;14222          ;011-----------------------------; 2ND PARTIAL PRODUCT
U 052B, 0C86,7009,0030,0047,005F,1              ;14223          M[TEMP7]_MB+Q                   ; GET THE SUM OF PART6 + PART 7
                                                ;14224 =
```

```
                                ;14225 =00
                                ;14226 =01
                                ;14227          ;01-------------------------------;
                                ;14228          M[TEMP0]_D+ALKC,                  ; PART 5 + CARRY SAVE IN TEMP0
U 05F1, 0C86,0061,003D,8047,046E,2  ;14229      PUSH,NEXT/FP.QT6.SC17             ; Q GETS TEMP6 RESET STEP TO 17
                                ;14230
                                ;14231 FP.MULD.SUB.20:
                                ;14232          ;10-------------------------------;
                                ;14233          MULFAST+ CAND IN R[TEMP2],        ; PART 2 AND 3 IN D & Q
                                ;14234          DBZ STEPC?,SIZE[LONG],            ; ( A AND D)
U 05F2, 0880,0027,9320,8047,005F,2  ;14235      NEXT/FP.MULD.SUB.20               ;
                                ;14236
                                ;14237          ;11-------------------------------; 3RD PARTIAL PRODUCT
U 05F3, 0480,7009,0030,0047,006E,4  ;14238      WB_M[TEMP7]+Q                     ; GET CARRY FROM PART4 +PREVIOUS VALUE
                                ;14239 =
                                ;14240          ;-------------------------------;
                                ;14241          M[TEMP7]_D+R[TEMP0]+ALKC,         ; PART 3 PLUS THE PREVIOUS CARRY
U 06E4, 0886,7061,0030,04E7,005F,5  ;14242      MDR_0                             ; CLOBBER MDR FOR NEXT INST
                                ;14243                                            ; AND SUM PART 5 AND PART 6
                                ;14244 =00
                                ;14245 =01
                                ;14246          ;01-------------------------------;
                                ;14247          MTEMP0_MDR+R[ZERO]+ALKC,          ; SAVE ALKC (MDR AND RSRC ARE ZERO)
                                ;14248          PUSH,                             ; RESET Q AND CHECK AGAIN FOR
U 05F5, 0087,2041,003D,8047,046E,3  ;14249      NEXT/FP.QT1.SC17.CHKINT           ; INTERRUPTS AND TIMER
                                ;14250
                                ;14251 FP.MULD.SUB.30:
                                ;14252          ;10-------------------------------;
                                ;14253          MULFAST+ CAND IN R[TEMP2],        ; Q HAS TEMP1 AND STEP = 17
                                ;14254          DBZ STEPC?,SIZE[LONG],            ; PART 1 AND PART 2 IN D AND Q
U 05F6, 0080,0027,9320,8047,005F,6  ;14255      NEXT/FP.MULD.SUB.30               ;
                                ;14256
                                ;14257          ;11-------------------------------;
                                ;14258          R[TEMP3]_M[TEMP7]+Q,              ; FINALLY BITS 32-63
U 05F7, 0C84,7009,00B0,C047,0000,1  ;14259      RETURN [+1]                       ; TEMP0 HAS PREVIOUS CARRY
                                ;14260                                            ; D AND ALKC HAVE BITS 0-31
```

```
;14261 .TOC "  Floating point and CRC          : FP.DIVD2 FP.DIVD3 "
;14262
;14263 ;*****************************************************************************
;14264 ;
;14265 ;               66      DIVD2   divr,rd,quo.md
;14266 ;
;14267 ;               67      DIVD3   divr.rd,divd.rd,quo.wd
;14268 ;
;14269 ;               Input:
;14270 ;                       Temp1                   divr.rd (bits 0-31)
;14271 ;                       MDR                     (bits 32-63)
;14272 ;
;14273 ;               Resources:
;14274 ;                       for divr
;14275 ;                       Temp2,Q                 fraction (bits 0-31)
;14276 ;                       Temp3                   fraction (bits 32-63)
;14277 ;                       Temp5                   exponent
;14278 ;                       Flag2                   sign of operand
;14279 ;                          (MUL1(FLAG2))
;14280 ;
;14281 ;               Subroutines:
;14282 ;                       FP.MULDOP1              parse divr
;14283 ;                       *OS.DMOD                get divd for DIVD2
;14284 ;                       *OS.DRED                get divd for DIVD3
;14285 ;                       OS.WRT1                 get quotient for DIVD3
;14286 ;                                             * return is through IRDROM to
;14287 ;                                             FP.DIVD.20 to continue operation
;14288 ;               Exit:
;14289 ;                       FP.WRITE.SEC            write bits 32-63 of answer
;14290 ;
;14291 ;               Operation:
;14292 ;                       These are the driving routines for the double precision
;14293 ;                       DIVD2 and DIVD3 instructions
;14294 ;
;14295 ;       **********FPA ENTRY AND INTERFACE**********
;14296 ;       DIVD2   (OS.RED) - FI.ADDD2.REG/FI.ADDD2.MEM
;14297 ;       DIVD3   (OS.RED) - FI.ADDD3.REG/FI.ADDD3.MEM
;14298 ;
;14299 ;*****************************************************************************
```

```
                                ;14300 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;14301 =00
                                ;14302 FP.DIVD2:                                ; **********************
                                ;14303          ;00-----------------------------;
                                ;14304          R[TEMP5]_EXP(M[TEMP1]),         ; PARSE DIVISOR
                                ;14305          FRO.FLTZ?,                      ; TEST VALIDITY OF OPERAND
U 00D4, 0484,1437,0AB1,4047,0C5E,4 323*  ;14306          PUSH,NEXT/FP.MULDOP1            ;
                                ;14307
                                ;14308          ;01-----------------------------;
                                ;14309          M[TEMP2]_Q,                     ; SAVE BITS 0-31 IN CORRECT TEMP
                                ;14310          LOD INC BRA?,                   ; GET DIVD RET THRU IRDROM TO FP.DIVD.20
U 00D5, 0086,203A,41BD,8047,0419,0       ;14311          PUSH,NEXT/OS.DMOD               ; FINAL RETURN HERE
                                ;14312
                                ;14313          ;10-----------------------------;
                                ;14314          R[DST.R]_M[TEMP2],              ; DO FIRST PART OF WRITE
                                ;14315          WRITE NOTREG,SIZE[IDEP],        ; NO INTERRUPTS ALLOWED
                                ;14316          SET MM.NOINT,CCOP1,             ; SET CONDITION CODES
U 00D6, 0064,2592,403C,4DDA,0056,F       ;14317          NEXT/FP.WRITE.SEC               ; CONTINUE THE WRITING
                                ;14318 =
                                ;14319 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;14320 =00
                                ;14321 FP.DIVD3:                                ; **********************
                                ;14322          ;00-----------------------------;
                                ;14323          R[TEMP5]_EXP(M[TEMP1]),         ; PARSE DIVISOR
                                ;14324          FRO.FLTZ?,                      ; TEST VALIDITY OF OPERAND
U 00D8, 0484,1437,0AB1,4047,0C5E,4 323*  ;14325          PUSH,NEXT/FP.MULDOP1            ;
                                ;14326
                                ;14327          ;01-----------------------------;
                                ;14328          M[TEMP2]_Q,                     ; Q GETS TO CORRECT TEMP (BITS0-31)
                                ;14329          PUSH,                           ; GET DIVD RET AT END
                                ;14330          LOD INC BRA?,                   ; RET NOW THR IRDDOM
U 00D9, 0886,203A,41BD,8047,0418,0       ;14331          NEXT/OS.DRED                    ; TO FP.DIVD.20
                                ;14332
                                ;14333          ;10-----------------------------;
                                ;14334          PUSH,                           ; GET DST FOR QUOT
                                ;14335          LOD INC BRA?,                   ;
U 00DA, 0C80,0036,41B0,0047,0414,0       ;14336          NEXT/OS.WRT1                    ;
                                ;14337
                                ;14338          ;11-----------------------------;
                                ;14339          R[DST.R]_M[TEMP2],              ; DO FIRST PART OF WRITE
                                ;14340          WRITE NOTREG,SIZE[IDEP],        ;
                                ;14341          SET MM.NOINT,                   ; NO INTERRUPTS ALLOWED
                                ;14342          CCOP1,                          ; SET CONDITION CODES
U 00DB, 0064,2592,403C,4DDA,0056,F       ;14343          NEXT/FP.WRITE.SEC               ; CONTINUE THE WRITING
```

```
;14344 .TOC "  Floating point and CRC          : FP.DIVD.20 "
;14345
;14346 ;*****************************************************************************
;14347 ;
;14348 ;
;14349 ;                FP.DIVD.20
;14350 ;
;14351 ;                        DOUBLE PRECISION DIVISION FOR DIVD2,DIVD3
;14352 ;
;14353 ;                        Input:
;14354 ;                                Temp1           dividend bits 0-31
;14355 ;                                MDR             bits 32-63
;14356 ;
;14357 ;                                for divisor
;14358 ;                                Temp2           fraction bits 0-31
;14359 ;                                Temp3           fraction bits 32-63
;14360 ;                                Temp5           exponent
;14361 ;                                Flag2           sign of divisor
;14362 ;                                   (MUL1(FLAG2))
;14363 ;                                Flag1           set if divisor 0
;14364 ;                                   (MOPZERO(FLAG1))
;14365 ;
;14366 ;                        Resources:
;14367 ;                                For divisor same as input
;14368 ;
;14369 ;                                for Dividend
;14370 ;                                Temp1           fraction bits 0-31
;14371 ;                                Temp6           fraction bits 32-63
;14372 ;                                RTemp8          exponent
;14373 ;                                Flag3           sign of dividend
;14374 ;                                   (MUL2(FLAG3))
;14375 ;                                D,Q             to perform division
;14376 ;                                Stepcounter     to control division loop
;14377 ;
;14378 ;               Subroutines:
;14379 ;                                FP.MULDOP2      to parse dividend
;14380 ;
;14381 ;
;14382 ;               Operation:
;14383 ;                        The dividend is parsed and validated
;14384 ;                        If  0 then the quotient is 0 and exit is made
;14385 ;                        If the divisor is 0 FP.DIVD.ZERO is entered
;14386 ;                        to make a fault exit to IE.FLDBZ.FAULT
;14387 ;                        Else the exponents are subtracted and rebiased
;14388 ;                        with the fudge factor of +62-hex for later
;14389 ;                        normalization
;14390 ;
;14391 ;                        The magnitude of the dividend is
;14392 ;                          D (bits 0-31)    Temp6 (bits 32-63)
;14393 ;
;14394 ;                        The magnitude of the divisor is
;14395 ;                          Temp2 (bits 0-31)  Temp3 (bits 32-63)
```

```
                                        ;14396 ;                                The  operation consists of a double length
                                        ;14397 ;                                compare and shift function using a nonrestoring
                                        ;14398 ;                                division operation
                                        ;14399 ;                                The operation is performed in 3 loops of 16,16,
                                        ;14400 ;                                26 iterations checks for interrupts and timer
                                        ;14401 ;                                are made at appropriate intervals
                                        ;14402 ;                                At the end of the total operation Temp2,Temp3
                                        ;14403 ;                                have the fraction The normalization
                                        ;14404 ;                                routine is entered at FP.NORM.MUDI
                                        ;14405 ;                                to determine the sign of the quotient by the
                                        ;14406 ;                                XOR of the operand signs Return is to the
                                        ;14407 ;                                calling routine if no floating overflow is
                                        ;14408 ;                                detected or floating underflow with FU bit set
                                        ;14409 ;                                FP.DIVD.ZERO
                                        ;14410 ;                                if the divisor is 0
                                        ;14411 ;                                exit is made to IE.FLDBZ.FAULT
                                        ;14412 ;
                                        ;14413 ;*********************************************************************************
                                        ;14414 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;14415 =00
                                        ;14416 FP.DIVD.20:
                                        ;14417         ;00-----------------------------; RET THRU IRDROM
                                        ;14418         R[TEMP8] EXP(M[TEMP1]),         ; TEST VALIDITY OF DIVD
                                        ;14419         FRO.FLTZ?,                      ;
U 00E8, 0084,1437,0AB2,0047,0C5E,8 323* ;14420         PUSH,NEXT/FP.MULDOP2            ;
                                        ;14421
                                        ;14422         ;01-----------------------------; RET +1 ANS IS EITHER
                                        ;14423         CLEAR MUL2(FLAG3),              ; CLEAR FLAG IF ANS IS ZERO
                                        ;14424         M[TEMP2] ZLIT12[8],             ; DIVIDE BY ZERO OR DIVIDEND IS 0
                                        ;14425         MOPZERO(FLAG1)?,                ; TEST MOPZERO(FLAG1) TO DIFFERENTIATE
U 00E9, 0D1E,2D77,05F0,4047,005F,8      ;14426         NEXT/FP.DIVD.ZERO               ;
                                        ;14427
                                        ;14428         ;10-----------------------------; RET +2 TIME TO DIVIDE
                                        ;14429         M[TEMP5] R[TEMP8]-M8,           ; SUBTRACT EXPONENTS
                                        ;14430         SIZE[WORD],PUSH,                ; OFF TO SUBRTN TO DO DIVIDE
U 00EA, 0086,5003,0012,0047,046E,5      ;14431         NEXT/FP.DIVD.45                 ;
                                        ;14432
                                        ;14433         ;11-----------------------------;
                                        ;14434         R[TEMP2] M[TEMP4],              ; RETURN TO WORKING TEMP
                                        ;14435         CLEAR FLAG0,                    ; CLEAR FLAG FOR OVERUNDER USE
                                        ;14436         MUL1(FLAG2) XOR MUL2(FLAG3)?,   ; SRC1 =0, 2XOR3 GIVES SIGN OF QUO
U 00EB, 0804,4592,4570,8047,0050,A      ;14437         NEXT/FP.NORM.MUDI               ; RETURN TO FIRST CALLING RTN
```

```
                                ;14438 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;14439 FP.DIVD.45:
                                ;14440          ;-------------------------------;
                                ;14441          M[TEMP5]_MB+ZLIT0[62],          ; TAKE CARE OF EXP FACTORS
U 06E5, 0186,5C11,0013,1047,006E,6  ;14442          SIZE[WORD]                      ;
                                ;14443
                                ;14444 FX.DIVD.50:                              ; USED BY G FLOAT
                                ;14445          ;-------------------------------;
U 06E6, 0D80,0EF6,4030,3047,006A,8  ;14446          PL_[6]                          ; SET UP PLATCH FOR LATER
                                ;14447
                                ;14448 =0
                                ;14449          ;0-------------------------------; SET UP D FOR DIVISION (BITS 0-31)
                                ;14450          D_M[TEMP1],CLEAR_FLAG0,         ; GET FLAG CLEAR FOR DIVISION LOOP
U 06A8, 0000,1002,603D,8047,046A,6  ;14451          PUSH,NEXT/FP.SC17               ; SET STEP TO 17 (ONE LARGER)
                                ;14452
                                ;14453          ;1-------------------------------; START OF MAIN DIVISION
                                ;14454          CLEAR_FLAG1,                    ; CLEAR FLAG FOR CONTROL SOON
U 06A9, 000E,6000,C030,C047,006A,A  ;14455          M[TEMP6]_(MB-R[TEMP3]).SL.1     ; DIVD_DIVD-DIVS (BITS 32-63)
                                ;14456                                          ; CARRY IN ZERO
                                ;14457 =0
                                ;14458 FP.DIVD.130:
                                ;14459          ;0-------------------------------;
                                ;14460          DIVDS SOR IN R[TEMP2],          ; DIVD (Q) - DIVS(TEMP2)
                                ;14461          ALUS_UNSGN,SIZE[LONG],          ; SAVE ALKC, STEP IS 1> AT FIRST TIME
U 06AA, 0080,0026,F320,90C7,006A,E  ;14462          DBZ STEPC?,NEXT/FP.DIVD.ALUS    ; THIS IS FOR LOOP
                                ;14463
                                ;14464          ;1-------------------------------;
                                ;14465          INTPEND OR TIMER?,              ; DO THE TEST HERE
U 06AB, 0480,0036,4AF0,0C47,04F7,0  ;14466          PUSH,NEXT/IE.SERV.IP.TS2        ; TEST FOR INT OR TIMER
```

```
                                ;14467 =01
                                ;14468 FP.DIVD.150:
                                ;14469          ;01-----------------------------, EQUIVALENT TO ALKC = 1
                                ;14470          M[TEMP6]_(MB-R[TEMP3]).SL.1,    ; SUBTRACT AND SHIFT
U 05F9, 0486,6000,CB30,D8E7,006A,A ;14471       IP.TS?,NEXT/FP.DIVD.130         ;
                                ;14472
                                ;14473          ;11-----------------------------; EQUIVALENT TO ALKC = 0
                                ;14474          M[TEMP6]_(MB+R[TEMP3]).SL.1,    ; ADD AND SHIFT
U 05FB, 0086,6001,CB30,D8E7,006A,C ;14475       IP.TS?                          ;
                                ;14476 =0
                                ;14477          ;0------------------------------;
                                ;14478          DIVDA SOR IN R[TEMP2],          ; DIVD + DIVS
                                ;14479          ALUS_UNSGN,SIZE[LONG],          ; SAVE IT
U 06AC, 0480,0027,F320,90C7,006A,E ;14480       DBZ STEPC?,NEXT/FP.DIVD.ALUS    ;
                                ;14481
                                ;14482          ;1------------------------------;
                                ;14483          INTPEND OR TIMER?,              ; DO THE TEST HERE
U 06AD, 0480,0036,4AF0,0047,04F7,0 ;14484       PUSH,NEXT/IE.SERV.IP.TS2        ; TEST FOR INT OR TIMER
                                ;14485
                                ;14486 =0
                                ;14487 FP.DIVD.ALUS:
                                ;14488          ;0------------------------------; STEP NE 0
                                ;14489          ALUS?,                          ; TEST ALKC SAVED AS NOTALKC
U 06AE, 0480,0036,4B70,1847,005F,9 ;14490       NEXT/FP.DIVD.150                ;
                                ;14491
                                ;14492          ;1------------------------------; STEP = 0
                                ;14493          M[TEMP0]_Q,                     ; SAVE QUOTIENT TEMP IN TEMP0
                                ;14494          FLAG<1-05>?,                    ; TEST WHICH TIME THROUGH LOOP
U 06AF, 0486,003A,453D,8047,005F,C ;14495       NEXT/FP.DIVD.NXT                ;
```

```
                                ;14496 =00
                                ;14497 FP.DIVD.NXT:
                                ;14498          ;00-------------------------------; FIRST TIME IN
                                ;14499          SET FLAG0,                        ; INDICATE THIS
                                ;14500          STEPC_ZLIT0[15.],                 ;
U 05FC, 0940,0C37,0030,7907,006A,E ;14501       NEXT/FP.DIVD.ALUS                 ; CONTINUE ON
                                ;14502
                                ;14503          ;01-------------------------------; SECOND TIME IN
                                ;14504          SET FLAG1,                        ; INDICATE THIS
U 05FD, 0948,0C37,0030,D107,005F,E ;14505       STEPC_ZLIT0[26.]                  ; SET STEP =26
                                ;14506
                                ;14507          ;10-------------------------------; CANT GET HERE WITH BRANCH
                                ;14508          R[TEMP4]_M[TEMP0],                ; SAVE FIRST PART (LOG OPERATION)
U 05FE, 0884,0592,4031,0047,006A,E ;14509       NEXT/FP.DIVD.ALUS                 ; ALUS OK
                                ;14510
                                ;14511          ;11-------------------------------; LAST TIME THROUGH
                                ;14512          M[TEMP3]_R[TEMP0].ASL.P,          ; SHIFT LEFT  PLATCH (6) BITS
U 05FF, 080E,3A37,00B0,0047,0000,1 ;14513       CLEAR FLAG1,RETURN [+1]           ;
                                ;14514
                                ;14515 =0*
                                ;14516 FP.DIVD.ZERO:
                                ;14517          ;0*-------------------------------; DIVIDEND IS 0
                                ;14518          CLEAR MUL1(FLAG2),                ; CLEAR OTHER ERROR FLAG
                                ;14519          R[TEMP2]_0,                       ;
U 05F8, 0C14,0587,00B0,8047,0000,1 ;14520       RETURN [+11                       ; MUST RETURN FOR CORRECT WRITE
                                ;14521
                                ;14522          ;1*-------------------------------; DIVISOR IS 0
U 05FA, 0480,0036,4030,0047,00FF,C ;14523       NEXT/IE.FLDBZ.FAULT               ; GO FAULT INSTRUCTION
```

```
;14524 .TOC "  Floating point and CRC          : FP.EMODF "
;14525
;14526 ;*****************************************************************************
;14527 ;
;14528 ;       54      EMODF   mulr.rf,mulrx.rb,muld.rf,int.wl,fract.wf
;14529 ;
;14530 ;                       Input:
;14531 ;                               Q               mulr.rf (multiplier)
;14532 ;                               MDR             mulrx.rb (multiplicand)
;14533 ;
;14534 ;                       Resources:
;14535 ;                               Temp2           fraction multiplier
;14536 ;                                               form bits 31-8 multiplier
;14537 ;                                                   bits 7-0 extension byte
;14538 ;                                               ie binary point before bit 31
;14539 ;                               Temp5           exponent
;14540 ;                               Flag2           several usages
;14541 ;                                 (MUL1(FLAG2)) sign of multiplier
;14542 ;                                  (OVER(FLAG2))        integer overflow
;14543 ;                               Temp0           fraction of multiplicand
;14544 ;                                               binary point before bit 31
;14545 ;                               RTemp8          exponent
;14546 ;                               Flag3           sign of multiplicand
;14547 ;                                 (MUL2(FLAG3))
;14548 ;                               D,Q             used for multiplication
;14549 ;                                               among other operations
;14550 ;                               Stepcounter     loop control of multiplication
;14551 ;                               MTemp8          save sign of product
;14552 ;                               RTemp10         RNUM/VA of int.wl
;14553 ;
;14554 ;                               Flag0           sign of product for cvt rtn
;14555 ;                                 (ADD1(FLAG0))
;14556 ;                               Flag1           several usuages
;14557 ;                                 (REGINT(FLAG1))  regmode for int.wl
;14558 ;                                 (MOPZERO(FLAG1))  either operand 0
;14559 ;                               MDR             integer
;14560 ;
;14561 ;                       Subroutines:
;14562 ;                               FP.MULFOP1      parse validate multiplier
;14563 ;                               FP.ZEROTEMP3    zero operand
;14564 ;                               OS.FRED         get mulr.rf (multiplicand)
;14565 ;                               FP.MULFOP2      parse validate multiplicand
;14566 ;                               OS.WRT1         get int.wl
;14567 ;                               OS.WRT1         get fract.wf
;14568 ;                               FP.CVTFI.SUBALT convert float to integer
;14569 ;                               FP.NORM.SNG     normalize,round,pack
;14570 ;                               MM.PRB.WRITE.SIZ.00 probe write access
;14571 ;
;14572 ;                       Operation:
;14573 ;                               Upon entry the multiplier and extension byte
;14574 ;                               are available FP.MULFOP1 parses and validates
;14575 ;                               the multiplier which is shifted left 1 and
;14576 ;                               merged with the extension byte - 32 bit value
```

```
;14577 ;                                OS.FRED fetches the multiplicand FP.MULFOP2
;14578 ;                                parses and validates which is also shifted
;14579 ;                                left 1 removing the overflow bit position
;14580 ;                                OS.WRT1 is called to get the integer (int.wl)
;14581 ;                                ******  the exponents are added, the sign of the
;14582 ;                                product determined If int.wl is in a register
;14583 ;                                RNUM is saved and REGINT(FLAG1) is set
;14584 ;                                otherwise VA is saved and the flag cleared
;14585 ;                                OS.WRT1 is called to fetch the fraction
;14586 ;                                (fract.wx) return is made to the calling rtn
;14587 ;                                unless the product was 0 in which case the
;14588 ;                                multiplication steps are effectively bypassed
;14589 ;                                regardless of the single double instruction
;14590 ;                                ******
;14591 ;                                        this routine FP.EMODF.250 is called also
;14592 ;                                        in processing EMODD instruction
;14593 ;                                *****
;14594 ;                                For EMODF the return is  FP.EMODF.30 to perform
;14595 ;                                the multiplication a 32 bit fraction
;14596 ;                                is produced & normalized to bit 31
;14597 ;                                *****
;14598 ;                                  Common flows for EMODD begin here
;14599 ;                                *****
;14600 ;                                If integer overflow or floating underflow
;14601 ;                                (FU bit not set) have occurred because of exp
;14602 ;                                manipulation, the integer and floating
;14603 ;                                operands are 0, otherwise a call is made to
;14604 ;                                FP.CVTFI.SUBALT to convert the value to an
;14605 ;                                integer with a floating fraction part
;14606 ;                                Upon return MDR contains the integer,
;14607 ;                                the fraction part is positioned with
;14608 ;                                the binary point before bit 31 and unnormalized
;14609 ;                                A fudge factor is added to the exponent
;14610 ;                                temp which is 0 except where the exp had been <0
;14611 ;                                in which case it contained the exp as computed
;14612 ;                                FP.NORM.SNG is branched to where the fraction
;14613 ;                                is normalized,rounded packed .
;14614 ;                                Upon return the fract.wx is tested for regmode
;14615 ;                                if not reg then a probe access test
;14616 ;                                is made at MM.PRB.WRITE.SIZ.00 . Upon return
;14617 ;                                or if regmode the int.wl is written from MDR
;14618 ;                                At this point the EMODF,EMODD instructions
;14619 ;                                diverge. the fract.wx is restored to VA/RNUM.
;14620 ;                                For each an error/nonerror path exist
;14621 ;                                if integer overflow the V bit is set
;14622 ;                                the low order bits of the true result have
;14623 ;                                been written out & IRD1 causes the trap
;14624 ;                                For non error the writes occur and IRD1 exit
;14625 ;
;14626 ;          **********FPA ENTRY AND INTERFACE**********
;14627 ;                  (OS.RED,OS.RED) - FI.EMODF
;14628 ;
;14629 ;*******************************************************************************
```

```
                                ;14630 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;14631 =0000
                                ;14632 =0001
                                ;14633          ;0001----------------------------; RET -1 FROM FP.MULFOP1
                                ;14634          MDR_0,                           ; ANSWER IS 0, MOPZERO(FLAG1) SET
U 01D1, 0C80,0036,4030,04E7,001D,3 ;14635       NEXT/FP.EMODF.10                 ;
                                ;14636
                                ;14637 FP.EMODF:                                 ; **********************
                                ;14638          ;0010----------------------------;
                                ;14639          M[TEMP2]_Q,                      ; MULTIPLIER TO AN MBUS SOURCE
                                ;14640          PUSH,                            ;
U 01D2, 0886,203A,403D,8047,0463,A ;14641       NEXT/FP.MULFOP1                  ; PARSE OPERAND
                                ;14642
                                ;14643 FP.EMODF.10:
                                ;14644          ;0011----------------------------; RETURN +1 FROM FP.MULFOP1
                                ;14645          R[TEMP2]_RB.ASL.1,               ; SHIFT MULTIPLIER LEFT 1 FOR EXTENSION
U 01D3, 0884,05F7,0010,8047,0454,2 ;14646       PUSH,NEXT/FP.ZEROTEMP3           ; ZERO SECOND HALF
                                ;14647
                                ;14648          ;0100----------------------------;
                                ;14649          R[TEMP2]_RB.OR.ZEXT(M[MDR]),     ; MERGE MULT AND EXTENSION
                                ;14650          SIZE[BYTE],                      ; TEMP2 TO GO TO RBUS
                                ;14651          LOD INC BRA?,                    ; GET THE MULTIPLICAND
U 01D4, 0085,2016,4180,8047,0416,0 ;14652       PUSH,NEXT/OS.FRED                ;
                                ;14653
                                ;14654          ;0101----------------------------;
                                ;14655          R[TEMP8]_EXP(M[MDR]),            ; EXTRACT EXPONENT OF MULTIPLICAND
                                ;14656          FRO.FLTZ?,                       ; TEST VALIDITY
U 01D5, 0C85,2437,0AB2,0047,0C5B,C 323* ;14657  PUSH,NEXT/FP.MULFOP2              ;
                                ;14658
                                ;14659          ;0110----------------------------;
                                ;14660          Q_M[TEMP0].SL.1,                 ; REMOVE THE OVERFLOW BIT (BP IS BIT 31)
                                ;14661          LOD INC BRA?,                    ; GET INT.WL DOES NOT DESTROY  Q
U 01D6, 0C80,0E52,D1B0,0047,0414,0 ;14662       PUSH,NEXT/OS.WRT1                ;
                                ;14663
                                ;14664          ;0111----------------------------;
                                ;14665          STEPC_ZLIT0[17.],                ; SET STEP COUNTER
U 01D7, 0580,0C37,0030,8907,046E,8 ;14666       PUSH,NEXT/FP.EMODF.210           ; RETURN HERE LATER
                                ;14667
                                ;14668 FP.EMODF.30:
                                ;14669          ;1000----------------------------; THE MULTIPLICATION
                                ;14670          MULFAST+ CAND IN R[TEMP2],       ; STEP IS 17 AT START
                                ;14671          SIZE[LONG],                      ;
                                ;14672          DBZ STEPC?,                      ; FINISHED THE LOOP ?
U 01D8, 0880,0027,9320,8047,001D,8 ;14673       NEXT/FP.EMODF.30                 ; LOOP TILL DONE
                                ;14674
                                ;14675          ;1001----------------------------; DONE
                                ;14676          M[TEMP2]_D,                      ; TEST IF PRODUCT IS NORMALIZED
                                ;14677          CLEAR MU[1(FLAG2),               ; CLEAR FLAG FOR LATER
U 01D9, 0816,25B2,46F0,0047,0060,1 ;14678       WB<31-30>?                       ;
                                ;14679 =
```

```
                                        ;14680 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                        ;14681 =01
                                        ;14682          ;01-----------------------------; NOT NORMALIZED
                                        ;14683          R[TEMP2]_M[TEMP2]+RB,           ; SHIFT LEFT FOR NORMALIZING
U 0601, 0084,2001,0030,8047,0061,2      ;14684          NEXT/FP.EMODFD.45               ;
                                        ;14685
                                        ;14686          ;11-----------------------------; NORMALIZED
                                        ;14687          M[TEMP5]_MB-ZLIT0[80],          ; REMOVE THE EXTRA BIAS
                                        ;14688          CLEAR MU[2(FLAG3),              ; CLEAR THIS FLAG ALSO
                                        ;14689          SIZE[WORD],SIGND CMP DEF?,      ; EXP >=0,<0
U 0603, 051E,5C10,0B54,0047,0853,1 413* ;14690          NEXT/FP.EMODF.60                ;
                                        ;14691
                                        ;14692 =00
                                        ;14693 FP.EMODF.50:
                                        ;14694          ;00-----------------------------; OVERFLOW INTEGER
                                        ;14695          R[TEMP2]_0,MDR_0,               ; EVERYTHING IS ALSO ZERO
                                        ;14696          SET OVER[FLAG2],                ; SET FLAG FOR LATER
U 0604, 0054,05B7,0030,84E7,00BF,1      ;14697          NEXT/FP.EMODF.55                ;
                                        ;14698
                                        ;14699          ;01-----------------------------;
                                        ;14700          WB_M[TEMP5].AND.ZLIT0[80],      ; MASK OUT ALL BUT BIAS BIT
                                        ;14701          WX.NE.0?,MDR_0,                 ; IS THE BIT SET
U 0605, 0180,5C12,0A74,04E7,046B,6      ;14702          PUSH,NEXT/FP.CVTFI.SUBALT       ; CONVERT FLOAT TO INTEGER
                                        ;14703
                                        ;14704          ;10-----------------------------;
U 0606, 0880,8002,603D,8047,0060,8      ;14705          D_M[TEMP8]                      ; RESTORE SIGN
                                        ;14706
                                        ;14707 =
                                        ;14708 =00
                                        ;14709          ;00-----------------------------; MDR HAS INTEGER NOT YET HANDLED SIGN
                                        ;14710          M[TEMP5]_MB+ZLIT0[61],          ; SET UP BIAS EXPONENT AND FACTOR (1F)
                                        ;14711          SIZE[WORD],                     ; FRACTION IN TEMP2,TEMP3
                                        ;14712          IR<5>?,                         ; IS IT SINGLE OR DOUBLE
U 0608, 0D86,5C11,08D3,0847,0468,C      ;14713          PUSH,NEXT/FP.NORM.SNG           ; GO NORMALIZE THE FRACTION
                                        ;14714
                                        ;14715          ;01-----------------------------; SAVE RNUM FROM FRACT.WX
                                        ;14716          R[TEMP1]_RNUM,REG MODE?,        ; WAS IT REG MODE
U 0609, 0085,6036,4930,4047,046B,0      ;14717          PUSH,NEXT/FP.EMODF.100          ;
                                        ;14718
                                        ;14719          ;10-----------------------------; RETURN ONLY IF EMODD
                                        ;14720          R[DST.R]_M[TEMP2],              ; WRITE FIRST PART
                                        ;14721          WRITE NOTREG,SET MM.NOINT,      ; NO INTERRUPTS
                                        ;14722          CCOP1,SIZE[IDEP],               ; CONDITION CODES
U 060A, 0064,2592,403C,4DDA,006E,7      ;14723          NEXT/FP.EMODF.125               ; FINISH UP
                                        ;14724 =
                                        ;14725 0BF1:
                                        ;14726 FP.EMODF.55:
                                        ;14727          ;*********FORCE ADDRESS*********;
U 0BF1, 0886,35B7,0030,0047,0053,4      ;14728          M[TEMP3]_0,NEXT/FP.EMODF.80     ; CLEAR BITS 63:32 IN FRACTION
```

```
                                ;14729 =000
                                ;14730 =001
                                ;14731 FP.EMODF.60:
                                ;14732          ;001----------------------------; EXP >=0
                                ;14733          WB_M[TEMP5].AND.ZLIT8[OFF],     ; SEE IF OVERFLOW
                                ;14734          SIZE[WORD],WX.EQ.0?,            ; IF NOT ZERO THEN OVERFLOW
U 0531, 0980,5D92,0A17,F847,0060,4  ;14735      NEXT/FP.EMODF.50                ;
                                ;14736
                                ;14737 =011
                                ;14738          ;011----------------------------; UNDERFLOW NO RETURN IF FU SET
                                ;14739          R[TEMP2] D_0,MDR_0,             ; ZERO OUT INT AND FRACTION
U 0533, 0C84,05B7,2030,84E7,045C,E  ;14740      PUSH,NEXT/FP.TESTPSL            ; SEE ABOUT FU BIT
                                ;14741
                                ;14742 FP.EMODF.80:
                                ;14743          ;100----------------------------; SAVE RNUM FROM FRACT.WX
                                ;14744          R[TEMP1] RNUM,REG MODE?,        ; WAS IT REG MODE
U 0534, 0085,6036,4930,4047,046B,0  ;14745      PUSH,NEXT/FP.EMODF.100          ;
                                ;14746
                                ;14747          ;101----------------------------; RETURN ONLY IF EMODD
                                ;14748          R[DST.R]_M[TEMP2],              ; WRITE FIRST PART
                                ;14749          WRITE NOTREG,SET MM.NOINT,      ; NO INTERRUPTS ALLOWED
                                ;14750          CCOP1,SIZE[IDEP],               ; SET CONDITION CODES
U 0535, 0064,2592,403C,4DDA,006E,7  ;14751      NEXT/FP.EMODF.125               ; FINISH UP EMODD
                                ;14752 =
```

```
                                ;14753 =0
                                ;14754 FX.EMODG.100:                            ; ENTRY FOR G FLOAT
                                ;14755 FP.EMODF.100:                            ;
                                ;14756          ;0------------------------------; NOT REG MODE
                                ;14757          R[TEMP1]_M[VA],PUSH,            ; SAVE VA INSTEAD
U 06B0, 0885,B592,4030,4047,0579,6  ;14758      NEXT/MM.PRB.WRITE.SIZ.00        ; GO PROBE SECOND OPERAND
                                ;14759
                                ;14760 FX.EMODH.INTWT:
                                ;14761          ;1------------------------------; REG MODE OR RETURN FROM PROBE
                                ;14762          VA_RNUM_R[TEMP10],              ; RESTORE INT.WL POINTERS
U 06B1, 0481,D5BE,4532,84A7,0060,C  ;14763      REGINT(FLAG1) ADD1(FLAG0)?      ; WAS IT REG MODE AND WHAT WAS SIGN
                                ;14764
                                ;14765 =00
                                ;14766          ;00-----------------------------; NOT REG MODE SIGN POSITIVE
                                ;14767          WRITE M[MDR],                   ; INTEGER IN MDR
                                ;14768          SET_MM.NOINT,SIZE[LONG],        ; NO INTERRUPTS NOW TILL DONE
U 060C, 0861,2592,48E0,05D8,006B,2  ;14769      IR<5>?,NEXT/FP.EMODF.120        ;
                                ;14770
                                ;14771          ;01-----------------------------; NOT REG MODE SIGN NEGATIVE
                                ;14772          WRITE -M[MDR],                  ; INTEGER IN MDR COMPLEMENT IT
                                ;14773          SET_MM.NOINT,SIZE[LONG],        ; NO INTERRUPTS NOW TILL DONE
U 060D, 0861,2003,08ED,85D8,006B,2  ;14774      IR<5>?,NEXT/FP.EMODF.120        ;
                                ;14775
                                ;14776          ;10-----------------------------; REG MODE SIGN POSITIVE
                                ;14777          R[GPR.R]_M[MDR],                ; WRITE TO THE REGISTER
                                ;14778          SET_MM.NOINT,SIZE[LONG],        ; NO INTERRUPTS TILL END
U 060E, 0C65,2592,48EC,C047,006B,2  ;14779      IR<5>?,NEXT/FP.EMODF.120        ;
                                ;14780
                                ;14781          ;11-----------------------------; REG MODE SIGN NEGATIVE
                                ;14782          R[GPR.R].SIZ_-M[MDR],           ; INTEGER IN MDR COMPLEMENT IT
                                ;14783          SET_MM.NOINT,SIZE[LONG],        ; NO INTERRUPTS TILL END
U 060F, 0C63,2593,08EC,C047,006B,2  ;14784      IR<5>?,NEXT/FP.EMODF.120        ;
                                ;14785
                                ;14786 =0
                                ;14787 FP.EMODF.120:
                                ;14788          ;0------------------------------; EMODF INSTRUCTION
                                ;14789          VA_RNUM_R[TEMP1],               ; RESTORE SECOND(FRACT.WX) OPERAND
                                ;14790          OVER(FLAG2)?,                   ; INTEGER OVERFLOW
U 06B2, 0081,D5BE,44B0,44A7,0063,3  ;14791      NEXT/FP.EMODF.WRITE             ;
                                ;14792
                                ;14793          ;1------------------------------; EMODD INSTRUCTION
                                ;14794          VA_RNUM_R[TEMP1],               ; RESTORE SECOND(FRACT.WX) OPERAND
U 06B3, 0C81,D5BE,40B0,44A7,0000,1  ;14795      RETURN [+1]                     ; FOR D OR G FLOAT
```

```
                                    ;14796 FP.EMODF.125:                          ; USED BY C FLOAT TO FINISH UP
                                    ;14797          ;-----------------------------;
                                    ;14798          VA_VA+4,                      ; TEST FOR ERRORS
U 06E7, 0C80,0036,44B0,0447,005F,0  ;14799          OVER(FLAG2)?                  ;
                                    ;14800 =0**
                                    ;14801          ;0**--------------------------; NORMAL EXIT
                                    ;14802          R[DST.R+1]_M[TEMP3],          ;
                                    ;14803          WRITE NOTREG,SIZE[LONG],      ; WRITE SECOND PART
U 05F0, 0884,3592,412F,45DA,003F,9  ;14804          IRD1                          ;
                                    ;14805
                                    ;14806          ;1**--------------------------; INTEGER OVERFLOW
                                    ;14807          R[DST.R+1]_M[TEMP3],          ;
                                    ;14808          WRITE NOTREG,SIZE[LONG],      ; WRITE SECOND PART
U 05F4, 0C84,3592,402F,45DA,0051,E  ;14809          NEXT/FP.SETVIRD1              ; SET V AND EXIT
                                    ;14810
                                    ;14811 =0**
                                    ;14812 FP.EMODF.WRITE:
                                    ;14813          ;0**--------------------------; EMODF
                                    ;14814          R[DST.R]_M[TEMP2],            ; NO ERROR PATH
                                    ;14815          WRITE NOTREG,SIZE[LONG],      ; WRITE THE FRACTION
U 0633, 0884,2592,412C,4DDA,003F,9  ;14816          CCOP1,IRD1                    ; SET CONDITION CODES
                                    ;14817
                                    ;14818          ;1**--------------------------; INTEGER OVERFLOW
                                    ;14819          R[DST.R]_M[TEMP2],            ; WRITE FRACTION
                                    ;14820          WRITE NOTREG,SIZE[LONG],      ;
                                    ;14821          CCOP1,                        ; SET CONDITION CODES
U 0637, 0C84,2592,402C,4DDA,0051,E  ;14822          NEXT/FP.SETVIRD1              ;
```

```
                                ;14823 FP.EMODF.210:
                                ;14824          ;-------------------------------;
                                ;14825          M[TEMP5]_MB+R[TEMP8],           ; ADD EXPONENTS
                                ;14826          SIZE[WORD],                     ;
                                ;14827          CLEAR ADD1(FLAG0),              ; CLEAR FLAG FOR USE LATER
                                ;14828          MOPZERO (MUL1 XOR MUL2)?,       ; MOPZERO(FLAG1)
                                ;14829                                          ; (MUL1(FLAG2) XOR MUL2(FLAG3))?,
U 06E8, 0006,5001,0552,0047,0053,8 ;14830       NEXT/FP.EMODF.250               ; SRC1=0,SIGN OF PRODUCT?
                                ;14831
                                ;14832 =000
                                ;14833 FP.EMODF.250:
                                ;14834          ;000----------------------------;
                                ;14835          M[TEMP8]_D_ZLIT0[0],            ; SIGN OF PRODUCT IS POSITIVE
                                ;14836          REG MODE?,                      ; WAS  INT.WL REGISTER?
U 0538, 0586,8C37,2930,0047,0053,E ;14837       NEXT/FP.EMODF.300               ; CONTINUE PROCESSING
                                ;14838
                                ;14839          ;001----------------------------;
                                ;14840          M[TEMP8]_D_ZLIT12[8],           ; SIGN OF PRODUCT IS NEGATIVE
                                ;14841          SET ADD1(FLAG0),                ; SET SIGN FLAG FOR CVT ROUTINE
                                ;14842          REG MODE?,                      ; WAS INT.WL REGISTER?
U 0539, 0146,8D77,2930,4047,0053,E ;14843       NEXT/FP.EMODF.300               ; CONTINUE PROCESSING
                                ;14844
                                ;14845          ;010----------------------------;
U 053A, 0C80,0036,4030,0047,0053,B ;14846       NEXT/FP.EMODF.255               ; ZERO INTEGER FRACTION IS 0
                                ;14847
                                ;14848 FP.EMODF.255:
                                ;14849          ;011----------------------------;
                                ;14850          R[TEMP2]_0,MDR_0,               ; ZERO INTEGER
                                ;14851          CLEAR MUL1(FLAG2),              ; NEED NOT CLEAR MOPZERO(FLAG1)
                                ;14852          PUSH,                           ; HANDLED AS REGINT(FLAG1)
                                ;14853          REG MODE?,                      ; WAS INT.WL REG MODE
U 053B, 0814,05B7,0930,84E7,0453,E ;14854       NEXT/FP.EMODF.300               ;
                                ;14855
                                ;14856          ;100----------------------------;
                                ;14857          R[TEMP1]_RNUM,MDR_0,            ; SAVE RNUM VALUE
                                ;14858          CLEAR MUL2(FLAG3),              ; CLEAR SECOND FLAG
U 053C, 081D,6036,4030,44E7,0053,4 ;14859       NEXT/FP.EMODF.80                ;
                                ;14860
                                ;14861 =0
                                ;14862 FX.EMODG.300:                            ; ENTRY FOR G FLOAT
                                ;14863 FI.EMODF.300:                            ; ENTRY FOR FPA INTERFACE
                                ;14864 FP.EMODF.300:                            ;
                                ;14865          ;0------------------------------; NOT REG MODE FOR INT.WL
                                ;14866          R[TEMP10]_M[VA],                ; SAVE VA FOR INT.WL
                                ;14867          CLEAR REGINT(FLAG1),            ; CLEAR FLAG INDICATING REGMODE
                                ;14868          LOD INC BRA?,                   ;
U 053E, 040D,B592,41B2,8047,0014,0 ;14869       NEXT/OS.WRT1                    ; GET FRACT.WX RETURN TO MAIN LINE
                                ;14870
                                ;14871          ;1------------------------------; REG MODE FOR INT.WL
                                ;14872          R[TEMP10]_RNUM,                 ; SAVE THE RNUM VALUE
                                ;14873          SET REGINT(FLAG1),              ; REG MODE FOR INT.WL
                                ;14874          LOD INC BRA?,                   ;
U 053F, 0C4D,6036,41B2,8047,0014,0 ;14875       NEXT/OS.WRT1                    ; GET FRACT.WX
```

```
;14876 .TOC "  Floating point and CRC          : FP.EMODD "
;14877
;14878 ;*************************************************************************
;14879 ;
;14880 ;       74      EMODD   mulr.rd,mulrx.rb,muld.rd,int.wl,fract.wd
;14881 ;
;14882 ;                       Input:
;14883 ;                               Temp1           mulr.rf bits 0-31
;14884 ;                               MDR             bits 32-63
;14885 ;                       Resources:
;14886 ;                                               for multiplier
;14887 ;                               Temp2           fraction bits 0-31
;14888 ;                               Temp3           fraction bits 32-63
;14889 ;                                                  all 64 bits of significance
;14890 ;                               Temp5           exponent
;14891 ;                                               for multiplicand
;14892 ;                               Temp1           fraction bits 0-31
;14893 ;                               Temp6           fraction bits 32-63
;14894 ;                               RTemp8          exponent
;14895 ;                               MDR             integer
;14896 ;                               Q               fraction then multiplication
;14897 ;                               Flag1           operand 0
;14898 ;                                  (MOPZERO(FLAG1))
;14899 ;                               Flag2           multiplier sign
;14900 ;                                  (MUL1(FLAG2))
;14901 ;                               Flag3           multiplicand sign
;14902 ;                                  (MUL2(FLAG3))
;14903 ;                               Stepcounter
;14904 ;                       Subroutines:
;14905 ;                               FP.MULDOP1              parse validate multiplier
;14906 ;                               OS.RED                  get mulrx.rb
;14907 ;                               FP.PREOPS.QT3           shift operands
;14908 ;                               OS.DRED                 get muld.rd
;14909 ;                               FP.MULDOP2              parse validate multiplicand
;14910 ;                               OS.WRT1                 get int.wl
;14911 ;                               FP.EMODF.250            get fract.wd
;14912 ;                               FP.PREOPS.SUBALT        prepare operands
;14913 ;                                                       call FP.MULD.SUB  to multiply
;14914 ;                       Exits
;14915 ;                               FP.EMODF.45             joins common flows with
;14916 ;                               FP.EMODF.60             EMODF instruction
;14917 ;                       Operation:
;14918 ;                               FP.MULDOP1 parses validates the multiplier.Extension
;14919 ;                               byte is fetched merged with the operand which has
;14920 ;                               been shifted left 1 .  The multiplicand is fetched
;14921 ;                               and parsed and validated .  The integer.wl is obtained
;14922 ;                               FP.EMODF.250 gets the fract.wd. FP.PREOPS.SUBALT
;14923 ;                               prepares the operands and calls FP.MULD.SUB to perform
;14924 ;                               the multiplication  The product is handled in
;14925 ;                               common flows with EMODF
;14926 ;
;14927 ;               **********FPA ENTRY AND INTERFACE**********
;14928 ;                       (OS.FIDRED,OS.RED) - FI.EMODD
;14929 ;
;14930 ;*************************************************************************
```

```
                                ;14931 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;14932 =0
                                ;14933 FP.EMODD:                                ; **********************
                                ;14934         ;0----------------------------;
                                ;14935         R[TEMP5] EXP(M[TEMP1]),         ; PARSE MULTIPLIER
                                ;14936         FRO.FLTZ?,                      ; TEST VALIDITY OF OPERAND
U 00D2, 0484,1437,0AB1,4047,0C5E,4 323*  ;14937         PUSH,NEXT/FP.MULDOP1            ;
                                ;14938
                                ;14939         ;1----------------------------;
                                ;14940         M[TEMP2] Q,                     ; SAVE BITS 0-31
                                ;14941         LOD INC BRA?,                   ; RET THRU IRDROM TO FP.EMODD.10
U 00D3, 0886,203A,41BD,8047,0010,0       ;14942         NEXT/OS.RED                     ;
                                ;14943
                                ;14944 =0000
                                ;14945 FP.EMODD.10:
                                ;14946         ;00C0--------------------------;
                                ;14947         MDR_ZEXT(M[MDR]),               ; ZERO EXT EXTENSION BYTE
                                ;14948         SIZE[BYTE],                     ;
U 02C0, 0081,2016,400D,8467,0467,0       ;14949         PUSH,NEXT/FP.PREOPS.QT3         ; SHIFT TEMP2 AND Q (GETS TEMP3)
                                ;14950
                                ;14951         ;0001--------------------------;
                                ;14952         R[TEMP3].SIZ_M[MDR].OR.Q,       ; MERGE BITS 32-63 WITH EXTENSION
                                ;14953         SIZE[LONG],                     ;
                                ;14954         PUSH,                           ; GET MULD.RD
                                ;14955         LOD INC BRA?,                   ;
U 02C1, 0C83,200A,41A0,C047,0418,0       ;14956         NEXT/OS.DRED                    ;
                                ;14957
                                ;14958         ;0010--------------------------;
                                ;14959         R[TEMP8] EXP(M[TEMP1]),         ; TEST VALIDITY OF MULD
                                ;14960         FRO.FLTZ?,                      ;
U 02C2, 0084,1437,0AB2,0047,0C5E,8 323*  ;14961         PUSH,NEXT/FP.MULDOP2            ;
                                ;14962
                                ;14963         ;0011--------------------------; RET +1 ANS IS ZERO
                                ;14964         R[TEMP2] 0,                     ; ZERO OUT FRACTION
                                ;14965         SET MOPZERO(FLAG1),             ;
U 02C3, 084C,05B7,0030,8047,002C,4       ;14966         NEXT/FP.EMODD.15                ; NOTHING TO DO BLESS THE COMMON FLOWS
                                ;14967
                                ;14968 FP.EMODD.15:
                                ;14969         ;0100--------------------------; RET +2
                                ;14970         STEPC_ZLIT0[17.],               ; BITS 0-31 IN TEMP1
                                ;14971         PUSH,                           ; BITS 32-63 IN TEMP6
                                ;14972         LOD INC BRA?,                   ; GO GET INT.WL
U 02C4, 0580,0C37,01B0,8907,0414,0       ;14973         NEXT/OS.WRT1                    ;
                                ;14974
                                ;14975         ;0101--------------------------;
                                ;14976         M[TEMP5] MB+R[TEMP8],           ; ADD EXPONENTS
                                ;14977         SIZE[WORD],                     ; SRC1=0 SIGN OF PRODUCT=XOR
                                ;14978         CLEAR ADD1(FLAG0),              ; CLEAR THIS FLAG FOR USE NOW
                                ;14979         PUSH,                           ; RET HERE AFTER NEXT OS.WRT1
                                ;14980         MOPZERO (MUL1 XOR MUL2)?,       ; MOPZERO(FLAG1)
                                ;14981                                         ; (MUL1(FLAG2) XOR MUL2(FLAG3));?,
U 02C5, 0806,5001,0552,0047,0453,8       ;14982         NEXT/FP.EMODF.250               ; JOIN EMODF CODE
```

```
                                        ;14983          ;0110----------------------------; MTEMP8 HAS SIGN BIT
                                        ;14984          Q_R[TEMP6],                       ; PUT Q INTO TEMP FOR SHIFTING
                                        ;14985          CLEAR MUL1(FLAG2),                ;
                                        ;14986          PUSH,                             ; SHIFT OPS AND MULTIPLY
U 02C6, 0010,05BE,5031,8047,0466,E      ;14987          NEXT/FP.PREOPS.SUBALT             ;
                                        ;14988
                                        ;14989          ;0111----------------------------;
                                        ;14990          CLEAR MUL2(FLAG3),                ; CLEAR FLAG FOR UNDER USE
                                        ;14991          M[TEMP2]_Q_D+R[TEMP0]+ALKC,       ; ADD CARRY TO BITS 0-31 PUT IN TEMP2
                                        ;14992          WB<31-30>?,                       ; NORMALIZED?
U 02C7, 041E,2061,16F0,0047,0861,1 338* ;14993          NEXT/FP.EMODD.40                  ;
                                        ;14994 =
                                        ;14995 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                        ;14996 =00
                                        ;14997 =01
                                        ;14998 FP.EMODD.40:
                                        ;14999          ;01----------------------------; NOT NORMALIZED
                                        ;15000          M[TEMP3]_(MB Q).RL.1,             ; ROTATE Q AND ALU LEFT 1
U 0611, 0086,3305,C03D,8047,0457,3      ;15001          PUSH,NEXT/FP.MT2Q                 ; PUT Q BACK TO TEMP2
                                        ;15002
                                        ;15003 FP.EMODFD.45:
                                        ;15004          ;10----------------------------;
                                        ;15005          M[TEMP5]_MB-ZLIT0[81],            ; MODIFY FOR EXP
                                        ;15006          CLEAR MUL2(FLAG3),                ; CLEAR THIS FLAG ALSO
                                        ;15007          SIZE[WORD],SIGND CMP DEF?,        ; EXP >=0,<0
U 0612, 011E,5C10,0B54,0847,0853,1 413* ;15008          NEXT/FP.EMODF.60                  ;
                                        ;15009
                                        ;15010 =11
                                        ;15011          ;11----------------------------;
                                        ;15012          M[TEMP5]_MB-ZLIT0[80],            ; REMOVE EXTRA BIAS
                                        ;15013          CLEAR MUL2(FLAG3),                ; CLEAR FLAG
                                        ;15014          SIZE[WORD],                       ;
                                        ;15015          SIGND CMP DEF?,                   ;
U 0613, 051E,5C10,0B54,0047,0853,1 413* ;15016          NEXT/FP.EMODF.60                  ;
```

```
;15017 .TOC "  Floating point and CRC          : FP.CVTFI.SUB FP.CVTFI.SUBALT "
;15018
;15019 ;*****************************************************************************
;15020 ;
;15021 ;       FLOATING TO INTEGER CONVERSION ROUTINE
;15022 ;
;15023 ;               FP.CVTFI.SUB - CVT INSTRUCTIONS
;15024 ;
;15025 ;               FP.CVTFI.SUBALT - EMODF EMODD INSTRUCTIONS
;15026 ;
;15027 ;               FX.CVTFI.SUBGC - CVT INSTRUCTIONS FOR G FLOAT
;15028 ;
;15029 ;               FX.CVTFI.SUBG - EMODG INSTRUCTION FOR G FLOAT
;15030 ;
;15031 ;                       Input:
;15032 ;                               Temp5           exponent
;15033 ;                               Temp2           fraction bits 0-31
;15034 ;                               Temp3           fraction bits 32-63 (or 0)
;15035 ;                                               binary point before bit 31
;15036 ;                               Flag0           sign of operand
;15037 ;                                 (ADDi(FLAG0))
;15038 ;                               Flag2           set = round for CVTRDL (-2**32)
;15039 ;                               Flag3           clear
;15040 ;                       Resources:
;15041 ;                               MDR             integer
;15042 ;                               Q               temp for range check
;15043 ;                               Flag2           set if integer overflow
;15044 ;                                 (OVER(FLAG2))
;15045 ;
;15046 ;                               Operation:
;15047 ;                                       A branching entry is made to differentiate
;15048 ;                                       between exponents <=,> 0.  If the exponent
;15049 ;                                       is < or = 0 the integer is 0  and the
;15050 ;                                       exponent is returned as is if <0.
;15051 ;                                       If in the range of 1-31 the 32 bit integer is
;15052 ;                                       extracted and the remaining fractional part
;15053 ;                                       correctly positioned in Temp2,Temp3 so that
;15054 ;                                       the binary point is before bit 31.
;15055 ;                                       If the exponent is 32 the integer part is
;15056 ;                                       extracted and a test made to determine
;15057 ;                                       if the value was the only non overflow
;15058 ;                                       case ie -2**32 . If not the overflow
;15059 ;                                       flag is set , the fractional
;15060 ;                                       part set up and a return effected
;15061 ;                                       For exponents > 32 overflow has
;15062 ;                                       definitely occurred and the remaining
;15063 ;                                       part of the routine is involved with
;15064 ;                                       finding the low order bits of the true
;15065 ;                                       integer result and positioning
;15066 ;                                       the fractional part accordingly
;15067 ;
;15068 ;*****************************************************************************
```

```
                                ;15069 FX.CVTFI.SUBGC:
                                ;15070          ;--------------------------------;
                                ;15071          WB_M[TEMP5].AND.ZLIT8[4],       ; MASK OUT ALL BUT BIAS BIT
                                ;15072          WX.NE.0?,MDR_0,                 ; IS THE BIT SET
U 06E9, 0D80,5D92,0A70,24E7,006B,4  ;15073          NEXT/FX.CVTFI.SUBG              ;
                                ;15074 =0
                                ;15075 FX.CVTFI.SUBG:
                                ;15076          ;0--------------------------------; EXP < 0
                                ;15077          Q_M[TEMP5]-ZLIT8[4],            ; PREPARE FOR EMODG  RETURN
                                ;15078          SIZE[WORD],MDR_0,               ; SET INTEGER TO 0
U 06B4, 0580,5D90,1490,24E7,006C,1  ;15079          FLAG2?,NEXT/FP.CVTFI.90         ; CHECK ON ROUND BIT CLEARED
                                ;15080
                                ;15081 FX.CVTFI.SUBG3:
                                ;15082          ;1--------------------------------; EXP > 0
                                ;15083          Q_ZLIT4[42]-M[MDR],             ; Q GETS BIAS AND 32 MDR HAS 0
U 06B5, 0D81,2DD3,1032,1047,006E,8  ;15084          NEXT/FP.CVTFI.ENT               ;
```

```
                                ;15085 FP.CVTFI.SUB:
                                ;15086          ;--------------------------------;
                                ;15087          WB_M[TEMP5].AND.ZLIT0[80],      ; MASK OUT ALL BUT BIAS BIT
U 06EA, 0980,5C12,0A74,04E7,006B,6 ;15088       WX.NE.0?,MDR_0                  ; IS THE BIT SET
                                ;15089
                                ;15090 =0
                                ;15091 FP.CVTFI.SUBALT:
                                ;15092          ;0--------------------------------; EXP < 0
                                ;15093          Q_M[TEMP5]-ZLIT0[80],           ; PREPARE FOR EMOD RETURN
                                ;15094          SIZE[WORD],MDR_0,               ; SET INTEGER TO 0
U 06B6, 0980,5C10,1494,04E7,006C,1 ;15095       FLAG2?,NEXT/FP.CVTFI.90         ; CHECK ON ROUND BIT CLEARED
                                ;15096
                                ;15097          ;1--------------------------------; EXP > 0
U 06B7, 0981,2C13,1035,0047,006E,8 ;15098       Q_ZLIT0[0A0]-M[MDR]             ; Q GETS BIAS AND 32 MDR HAS 0
                                ;15099
                                ;15100 FP.CVTFI.ENT:
                                ;15101          ;--------------------------------;
                                ;15102          PL_Q-M[TEMP5],                  ; LOAD UP PLATCH WITH EXPONENT DIFF
                                ;15103          MDR_0,                          ; SET UP FOR LATER
U 06EB, 0480,5BC8,0DF0,04E7,0861,4 376* ;15104  EXPONENT RANGE?                  ; WHAT IS EXP RANGE
                                ;15105
                                ;15106 =00
                                ;15107          ;00-------------------------------; WB <1-31> EXP
                                ;15108          MDR_(M[MDR] R[TEMP2]).RR.P,     ; GET INTEGER
                                ;15109          CLEAR FLAG2,                    ; BE SURE FLAG IS CLEAR
U 0614, 0411,2137,08B0,8467,006B,8 ;15110       IR<2>?,NEXT/FP.CVTFI.30         ; EMODX OR CVTS? OR G FLOAT TYPES
                                ;15111
                                ;15112          ;01-------------------------------; WB=0 EXP 32
                                ;15113          MDR_R[TEMP2],                   ;
                                ;15114          ADDT(FLAG0)?,                   ; VALUE NEGATIVE?
U 0615, 0C80,05BE,4430,8467,006B,A ;15115       NEXT/FP.CVTFI.35                ; ZERO OUT THE EXP FOR EMODX
                                ;15116
                                ;15117          ;10-------------------------------; WB=32 EXP =0
                                ;15118          R[TEMP5]_0,MDR_0,               ; INTEGER IS 0
                                ;15119          CLEAR FLAG2,                    ; CLEAR FLAG MAY BE USED FOR ROUND
U 0616, 0014,05B7,00B1,44E7,0000,1 ;15120       RETURN [+1]                     ;
                                ;15121
                                ;15122          ;11-------------------------------; WB >32 EXP >32
U 0617, 0580,0C37,2031,0047,006E,C ;15123       D_ZLIT0[20]                     ; SET UP TO  INCREMENT Q
                                ;15124
                                ;15125          ;--------------------------------;
                                ;15126          Q_D+Q,                          ; SUB FROM 32 AND BIAS
U 06EC, 0850,0029,1030,0047,006E,D ;15127       SET OVER(FLAG2)                 ; OVERFLOW TIME
                                ;15128
                                ;15129          ;--------------------------------;
                                ;15130          PL_Q-M[TEMP5],                  ; SET UP PLATCH WITH DIFF
                                ;15131          EXPONENT RANGE?,                ; MORE TESTING
U 06ED, 0480,5BCB,0DF0,0047,0854,0 376* ;15132  NEXT/FP.CVTFI.50                 ;
```

```
                                ;15133 =0
                                ;15134 FP.CVTFI.30:
                                ;15135          ;0------------------------------; FOR CVT INSTRUCTIONS
                                ;15136          M[TEMP2]_(MB R[TEMP3]).RR.P,    ; GET BITS 0-31 OF FRACTION
U 06B8, 0486,2137,0030,C047,006E,E ;15137       NEXT/FP.T3ZERO                  ;
                                ;15138
                                ;15139          ;1------------------------------; FOR EMODX INSTRUCTIONS
                                ;15140          R[TEMP5]_0,                     ; OR GFLOAT INSTRUCTIONS
U 06B9, 0484,05B7,0031,4047,006B,8 ;15141       NEXT/FP.CVTFI.30                ;
                                ;15142
                                ;15143 FX.T3ZERO:                               ; USED BY G FLOAT TYPE
                                ;15144 FP.T3ZERO:
                                ;15145          ;------------------------------;
                                ;15146          M[TEMP3]_(MB R[ZERO]).RR.P,     ; GET BITS 32-63
U 06EE, 0C86,3137,00BD,8047,0000,1 ;15147       RETURN [+1]                     ;
                                ;15148 =0
                                ;15149 FP.CVTFI.35:
                                ;15150          ;0------------------------------;
                                ;15151          R[TEMP2]_M[TEMP3],              ; VALUE POSITIVE OVERFLOW TIME
                                ;15152          SET OVER(FLAG2),                ;
U 06BA, 0054,3592,4030,8047,0053,7 ;15153       NEXT/FP.CVTFI.75                ; FINISH UP
                                ;15154
                                ;15155          ;1------------------------------; VALUE NEGATIVE MORE TESTING FOR -2**32
                                ;15156          R[TEMP5]_0,                     ; VALUE NEGATIVE MORE TESTING
U 06BB, 0484,05B7,04B1,4047,006C,0 ;15157       FLAG2?                          ; ROUND OR TRUNCATE?
                                ;15158
                                ;15159 =0**
                                ;15160          ;0**-----------------------------; TRUNCATE
                                ;15161          R[TEMP2]_RB.ASL.1,              ; SHIFT OFF KNOWN BIT 31
                                ;15162          WX.EQ.0?,                       ; WAS IT OK CASE
U 06C0, 0084,05F7,0A10,8047,086B,C 337* ;15163  NEXT/FP.CVTFI.45                 ;
                                ;15164
                                ;15165          ;1**-----------------------------; ROUND
U 06C4, 0910,0EF6,4030,F847,006E,F ;15166       PL_[1F],CLEAR OVER(FLAG2)       ; CHECK FOR THE OVERFLOW CASE
                                ;15167
                                ;15168          ;------------------------------;
                                ;15169          M[TEMP2]_(MB R[TEMP3]).RR.P,    ; GET THE 32 BITS OF INTEREST
U 06EF, 0486,2137,0A30,C047,086B,C 323* ;15170  WX.EQ.0?                         ;
                                ;15171
                                ;15172 =0
                                ;15173 FP.CVTFI.45:
                                ;15174          ;0------------------------------; THIS IS NOT GOOD OVERFLOW TIME
                                ;15175          R[TEMP2]_M[TEMP3],              ; SET UP FRACTION BITS 0-31
                                ;15176          SET OVER(FLAG2),                ; SET THE OVERFLOW FLAG
U 06BC, 0854,3592,4030,8047,0054,2 ;15177       NEXT/FP.ZEROTEMP3               ;
                                ;15178
                                ;15179 FX.T3TOT2:
                                ;15180 FP.T3TOT2:
                                ;15181          ;1------------------------------; THIS IS -2**32 OK
                                ;15182          R[TEMP2]_M[TEMP3],              ; SET UP FRACTION BITS 0-31
U 06BD, 0084,3592,4030,8047,0054,2 ;15183       NEXT/FP.ZEROTEMP3               ; THIS IS USED ALSO TO MOVE T3 TO T2
```

```
                                ;15184 =000
                                ;15185 FP.CVTFI.50:
                                ;15186          ;000-----------------------------; WB 1-31 EXP 33-63
                                ;15187          MDR_(M[TEMP2] R[TEMP3]).RR.P,    ; INTEGER TEMP IN MDR
U 0540, 0480,2137,0030,C467,006B,E ;15188       NEXT/FP.CVTFI.55                 ;
                                ;15189
                                ;15190          ;001-----------------------------; WB = 0 EXP 64
                                ;15191          MDR_R[TEMP3],                    ; MOVE THE INTEGER DIRECTLY
U 0541, 0C80,05BE,4030,C467,0468,E ;15192       PUSH,NEXT/FP.ZEROTEMP2           ; GO ZERO BOTH PARTS FRACTION
                                ;15193
                                ;15194 FX.ZEROTEMP3:
                                ;15195 FP.ZEROTEMP3:
                                ;15196          ;010-----------------------------; EXP = 32 ALREADY FILTERED
                                ;15197          R[TEMP3]_0,                      ; USE FOR PREVIOUS WORD
U 0542, 0884,05B7,00B0,C047,0000,1 ;15198       RETURN [+1]                      ; ZERO OUT TEMP3
                                ;15199
                                ;15200          ;011-----------------------------;
                                ;15201          Q_D+Q,                           ; BUMP UP Q AGAIN
U 0543, 0880,0029,1030,0047,0468,E ;15202       PUSH,NEXT/FP.ZEROTEMP2           ;
                                ;15203
                                ;15204          ;100-----------------------------;
                                ;15205          PL_Q-M[TEMP5],MDR_0,             ; PLATCH GETS DIFF
                                ;15206          EXPONENT RANGE?,                 ; INTEGER ZEROED FOR >95 EXPONENT
U 0544, 0480,5BCB,0DF0,04E7,0853,6 376* ;15207  NEXT/FP.CVTFI.70                 ;
                                ;15208 =
                                ;15209 =0
                                ;15210 FP.CVTFI.55:
                                ;15211          ;0-----------------------------;
U 06BE, 0886,3137,003D,8047,006B,F ;15212       M[TEMP3]_(MB R[ZERO]).RR.P       ; GET BITS 0-31 OF FRACTION
                                ;15213
                                ;15214          ;1-----------------------------;
                                ;15215          R[TEMP5]_0,                      ; ZERO OUT EXPONENT
J 06BF, 0C84,05B7,0031,4047,006B,C ;15216       NEXT/FP.CVTFI.45                 ; CONTINUE ENDUP
```

```
                                ;15217 =10
                                ;15218 FP.CVTFI.70:
                                ;15219          ;10-----------------------------;
U 0536, 0880,3137,003D,8467,0053,7  ;15220          MDR_(M[TEMP3] R[ZERO]).RR.P     ; GET INTEGER BITS
                                ;15221
                                ;15222 FP.CVTFI.75:
                                ;15223 FX.CVTFI.75:
                                ;15224          ;11-----------------------------;
                                ;15225          R[TEMP5]_0,                     ; ZERO OUT EXPONENT
U 0537, 0484,05B7,0031,4047,0054,2  ;15226          NEXT/FP.ZEROTEMP3
                                ;15227
                                ;15228 FX.ZEROTEMP1:
                                ;15229 FP.ZEROTEMP1:
                                ;15230          ;-----------------------------;
                                ;15231          R[TEMP1]_0,                     ; ZERO OUT FRACTION
U 06F0, 0C84,05B7,00B0,4047,0000,1  ;15232          RETURN [+1]                     ;
                                ;15233
                                ;15234 =0**
                                ;15235 FP.CVTFI.90:
                                ;15236 FX.CVTFI.90:
                                ;15237          ;0**----------------------------;
                                ;15238          M[TEMP5]_Q,                     ; FOR EMODS AND CVTS OTHER THAN RFL,RDL
U 06C1, 0886,503A,40BD,8047,0000,1  ;15239          RETURN [+1]                     ;
                                ;15240
                                ;15241          ;1**----------------------------;
                                ;15242          R[TEMP2]_0,CLEAR FLAG2,         ; CLEAR ROUND POSITION
U 06C5, 0C14,05B7,00B0,8047,0000,1  ;15243          RETURN [+1]                     ; FOR CVTRFL,RDL
```

```
;15244 .TOC "  Floating point and CRC          : FP.ACBF "
;15245
;15246 ;*****************************************************************************
;15247 ;
;15248 ;       4F      ACBF    limit.rf,add.rf,index.mf,displ.bw
;15249 ;
;15250 ;                       Input:
;15251 ;                               Q               limit.rf
;15252 ;                               MDR             addend.rf
;15253 ;                       Resources:
;15254 ;                               MTemp9          saves limit
;15255 ;                               For addend
;15256 ;                               **Temp2         fraction
;15257 ;                               Temp5           exponent
;15258 ;                               Flag0           sign of addend
;15259 ;                                  (ADD1(FLAG0))
;15260 ;                               For index
;15261 ;                               **Temp0         fraction
;15262 ;                               Temp7           exponent
;15263 ;                               Flag1           sign of index
;15264 ;                                  (ADD2(FLAG1))
;15265 ;                                       ** these temps have different contents
;15266 ;                                         on return from the FP.ADDRTN routine
;15267 ;                               Q               limit packed rotated for comparison
;15268 ;                               D               new index rotated for comparison
;15269 ;                               PC              may be incremented by displacement
;15270 ;                       Subroutines:
;15271 ;                               OS.MOD          get index
;15272 ;                               FP.ADDFOP1      parse addend
;15273 ;                               FP.ADDFOP2      parse index call FP.ADDRTN to
;15274 ;                                               perform add normalize and round and pack
;15275 ;                                               routines also called
;15276 ;                       Operation:
;15277 ;                               Both addend and index are parsed and validated
;15278 ;                               The add function is performed in common
;15279 ;                               flows at FP.ADDRTN.  Return is with new index
;15280 ;                               packed in Temp2.  The displacement is
;15281 ;                               fetched from the XB and the new index written out
;15282 ;                               No return if floating underflow (FU bit set)
;15283 ;                               or floating overflow detected in normalization routine
;15284 ;                               Otherwise the sign of the addend is checked to
;15285 ;                               determine the branching test
;15286 ;                               If the addend >=0
;15287 ;                                       index <= limit branch
;15288 ;                               if the addend < 0
;15289 ;                                       index >= limit branch
;15290 ;                               After adjusting the PC if appropriate exit is made
;15291 ;
;15292 ;       **********NOT AN FPA INSTRUCTION**********
;15293 ;
;15294 ;*****************************************************************************
```

```
                                ;15295 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;15296 =000
                                ;15297 FP.ACBF:                                  ; **********************
                                ;15298          ;000-----------------------------; LIMIT IN Q , ADD IN MDR
                                ;15299          M[TEMP9]_Q,                      ; SAVE LIMIT IN MTEMP9
                                ;15300          PUSH,LOD INC BRA?,               ; GET THIRD OPERAND
U 00E0, 0C86,903A,41BD,8047,0411,0  ;15301      NEXT/OS.MOD                      ;
                                ;15302
                                ;15303          ;001-----------------------------; ADD MOVED TO Q INDEX IN MDR
                                ;15304          M[TEMP2]_Q,                      ; ADD TO BE PARSED
U 00E1, 0086,203A,403D,8047,045A,1  ;15305      PUSH,NEXT/FP.ADDFOP1             ; TEST VALIDITY OF OPERAND
                                ;15306
                                ;15307          ;010-----------------------------; TEST VALIDITY OF INDEX
                                ;15308          R[TEMP7]_EXP(M[MDR]),            ; RETURN ONLY AFTER HAVE ANSWER
                                ;15309          FRO.FLTZ?,                       ; TEMP2 HAS NEW INDEX
U 00E2, 0C85,2437,0AB1,C047,0C59,8 323*  ;15310  PUSH,NEXT/FP.ADDFOP2             ; ADD1(FLAGO) SET TO REFLECT ADDEND
                                ;15311
                                ;15312          ;011-----------------------------;
                                ;15313          WB_EXP(M[TEMP9]),                ; TEST VALIDITY OF LIMIT BUT DONT UNPACK
                                ;15314          FRO.FLTZ?,                       ; Q GETS THE ROTATED VERSION OF LIMIT
U 00E3, 0C80,9437,0AB0,0047,0C61,C 323*  ;15315  PUSH,NEXT/FP.ACBF.500            ; FOR COMPARISON
                                ;15316
                                ;15317          ;100-----------------------------;
                                ;15318          R[DST.R]_M[TEMP2],               ;
                                ;15319          WRITE NOTREG,SIZE[LONG],         ; WRITE OUT NEW INDEX
                                ;15320          SET MM.NOINT,                    ; NO INTERRUPTS TILL END OF INSTRUCTION
                                ;15321          ADD1(FLAGO)?,                    ; SIGN OF ADD.RF
U 00E4, 0C64,2592,442C,45DA,006C,2  ;15322      NEXT/FP.ACBF.400                 ;
                                ;15323 =
                                ;15324 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;15325 =0
                                ;15326 FP.ACBF.400:
                                ;15327          ;0-----------------------------; ADD.RF IS POSITIVE
                                ;15328          WB_Q-D,                          ; LIMIT-INDEX
                                ;15329          SIGND CMP?,SIZE[LONG],           ; OR INDEX-LIMIT IF BOTH NEG
U 06C2, 0480,002B,0B60,0047,0861,9 368*  ;15330  NEXT/FP.ACBF.450                 ; TESTING FOR BRANCH
                                ;15331
                                ;15332          ;1-----------------------------; ADD.RF IS NEGATIVE
                                ;15333          WB_D-Q,                          ; INDEX-LIMIT
                                ;15334          SIGND CMP?,SIZE[LONG],           ; OR LIMIT -INDEX IF BOTH NEG
U 06C3, 0480,0028,0B60,0047,0861,9 368*  ;15335  NEXT/FP.ACBF.450                 ; TESTING FOR BRANCH
                                ;15336
                                ;15337 =01
                                ;15338 FP.ACBF.450:
                                ;15339          ;01----------------------------; >=0 BRANCH CONDITION
U 0619, 0080,0002,403D,8587,0061,B  ;15340      PC_PC+M[TEMPO]                   ; BUMP PC BY DISPLACEMENT
                                ;15341
                                ;15342          ;11----------------------------; <0 NO BRANCH CONDITION
U 061B, 0080,0036,4130,0047,003F,9  ;15343      IRD1                             ; EXIT
```

```
                                        ;15344 =00
                                        ;15345 FX.ACBG.500:
                                        ;15346 FP.ACBF.500:
                                        ;15347          ;00-------------------------------; RESERVED OPERAND ON LIMIT
U 061C, 0C80,0036,4030,0047,00FF,8      ;15348          NEXT/IE.OPER.FAULT                ;
                                        ;15349
                                        ;15350          ;01-------------------------------; LIMIT < 0
                                        ;15351          D_M[TEMP2].RR.16,                 ; ROTATE INDEX FOR COMPARISON
                                        ;15352          WB<31-30>?,                       ; IS INDEX ALSO NEGATIVE
U 061D, 0C80,23B7,26D0,0047,0862,1 337* ;15353          NEXT/FP.ACBF.550                  ;
                                        ;15354
                                        ;15355          ;10-------------------------------; LIMIT = 0
                                        ;15356          R[TEMP8]_Q_0,                     ; ZERO OUT OPERANDS USED FOR ACBD
                                        ;15357          CLEAR FLAGT,                      ; FLAG USED FOR SECOND HALF ACBD
U 061E, 0C0C,05B7,1032,0047,006F,1      ;15358          NEXT/FP.ACBF.525                  ; ROTATE INDEX AND RETURN
                                        ;15359
                                        ;15360          ;11-------------------------------; LIMIT > 0
                                        ;15361          CLEAR FLAG1,                      ; FLAG USED FOR SECOND HALF ACBD
U 061F, 0808,93B7,1010,0047,006F,1      ;15362          Q_M[TEMP9].RR.16                  ; ROTATE LIMIT FOR COMPARISON
                                        ;15363
                                        ;15364 FP.ACBF.525:
                                        ;15365          ;-------------------------------;
                                        ;15366          D_M[TEMP2].RR.16,                 ; ROTATE INDEX FOR COMPARISON
U 06F1, 0480,23B7,2010,0047,006F,3      ;15367          NEXT/FP.ACBF.570                  ; GET DISPLACEMENT
                                        ;15368 =01
                                        ;15369 FP.ACBF.550:
                                        ;15370          ;01-------------------------------; INDEX IS POSITIVE
                                        ;15371          Q_M[TEMP9].RR.16,                 ; ROTATE THE LIMIT FOR COMPARISON
U 0621, 0080,93B7,1010,0047,006F,3      ;15372          NEXT/FP.ACBF.570                  ; GET DISPLACEMENT
                                        ;15373
                                        ;15374          ;11-------------------------------; INDEX IS NEGATIVE ALSO
                                        ;15375          Q_M[TEMP9].RR.16,                 ; ROTATE THE LIMIT FOR COMPARISON
U 0623, 0048,93B7,1010,0047,006F,2      ;15376          SET FLAG1                         ; SET FLAG FOR SECOND HALF OF ACBD
                                        ;15377
                                        ;15378          ;-------------------------------;
                                        ;15379          D_Q_Q_D,                          ; INVERT THE TWO OPERANDS
U 06F2, 0480,002D,7030,0047,006F,3      ;15380          NEXT/FP.ACBF.570                  ; GET DISPLACEMENT
```

```
                                        ;15381 ;=0
                                        ;15382 06F3:    ;*********FORCE ADDRESS*********; NEED NOT BE 0 JUST SEQ MODE 64
                                        ;15383 FP.ACBF.570:
                                        ;15384          ;0-------------------------------;
                                        ;15385          M[TEMP8] PSL,                   ; GET THE C BIT TO SAVE
U 06F3, 0886,8036,4030,0087,0471,F      ;15386          PUSH,NEXT/FP.ACBF.580           ; GET THE DISPL FROM XB
                                        ;15387
                                        ;15388 06F4:    ;1-------------------------------;
                                        ;15389          M[TEMP8] MB.AND.ZLITO[1],       ; STRIP TO C BIT ONLY
U 06F4, 0986,8C12,0470,0847,006C,6      ;15390          GFLOAT(FLAG4)?                  ; WAS IT GH TYPE
                                        ;15391 =0
                                        ;15392          ;0-------------------------------; ACBF OR ACBD
                                        ;15393          WB_EXP(M[TEMP2]),               ; GET THE SETTING FOR CCS
                                        ;15394          FRO.FLTZ?,                      ;
U 06C6, 0880,2437,0AB0,0047,0862,4 323* ;15395          NEXT/FP.VALIDCC                 ; RETURN TO CALLER
                                        ;15396
                                        ;15397          ;1-------------------------------; ACBG OR ACBH
                                        ;15398          WB_M[TEMP2].XZ.MM,              ; SET THE CCS
                                        ;15399          CLEAR GFLOAT(FLAG4),            ; PL/SL MUST BE SET FOR G OR H
                                        ;15400          FRO.FLTZ?,                      ;
U 06C7, 0820,2077,0AB0,0047,0862,4 335* ;15401          NEXT/FP.VALIDCC                 ; RETURN IS TO CALLER
                                        ;15402
                                        ;15403 071F:    ;*********FORCE ADDRESS*********;
                                        ;15404 FP.ACBF.580:
                                        ;15405          ;-------------------------------; GET DISPL FROM XB
                                        ;15406          MTEMP0_SEXT(XB)+R[ZERO] PC_PC+2,; BUMP UP PC ALSO
U 071F, 0087,7815,009D,A047,0000,1      ;15407          RETURN [+1]                     ; RETURN TO CALLING ROUTINE
                                        ;15408
                                        ;15409
                                        ;15410 =00
                                        ;15411 FX.VALIDCC:                              ; USE FOR GH FLOAT
                                        ;15412 FP.VALIDCC:                              ;
                                        ;15413          ;00------------------------------; RESERVED OPERAND
U 0624, 0C80,0036,4030,0047,00FF,8      ;15414          NEXT/IE.OPER.FAULT              ;
                                        ;15415
                                        ;15416          ;01------------------------------; NEW INDEX < 0
                                        ;15417          CC_M[TEMP8].OR.ZLITO[8],        ;
U 0625, 0980,8C12,40B0,40A7,0000,1      ;15418          RETURN [+1]                     ;
                                        ;15419
                                        ;15420          ;10------------------------------; NEW INDEX = 0
                                        ;15421          CC_M[TEMP8].OR.ZLITO[4],        ;
U 0626, 0580,8C12,40B0,20A7,0000,1      ;15422          RETURN [+1]                     ;
                                        ;15423
                                        ;15424          ;11------------------------------; NEW INDEX > 0
                                        ;15425          CC_M[TEMP8].OR.ZLITO[0],        ;
U 0627, 0D80,8C12,40B0,00A7,0000,1      ;15426          RETURN [+1]                     ;
```

```
                        ;15427 .TOC "  Floating point and CRC          : FP.ACBD "
                        ;15428
                        ;15429 ;*****************************************************************************
                        ;15430 ;
                        ;15431 ;       6F      ACBD    limit.rd,add.rd,index.md,displ.bw
                        ;15432 ;
                        ;15433 ;                       Input:
                        ;15434 ;                               Temp1           limit bits 0-31
                        ;15435 ;                               MDR             bits 32-63
                        ;15436 ;                       Resources:
                        ;15437 ;                               MTemp9          limit packed bits 0-31
                        ;15438 ;                               Rtemp8          bits 32-63
                        ;15439 ;                               Temp0           displacement
                        ;15440 ;                               Q               limit rotated  for testing
                        ;15441 ;                               D               index rotated for testing
                        ;15442 ;                       Subroutines:
                        ;15443 ;                               OS.DRED         add.rd return through
                        ;15444 ;                                               IRDROM to FP.ACBD.100
                        ;15445 ;                                               which does add,normalize pack
                        ;15446 ;                                               operations before returning
                        ;15447 ;
                        ;15448 ;                       Operation:
                        ;15449 ;                               The limit is saved,OS.DRED gets the addend.rd
                        ;15450 ;                               return is through IRDROM to FP.ACBD.100
                        ;15451 ;                               to get index parse validate it and perform the
                        ;15452 ;                               add operation and then normalize round and
                        ;15453 ;                               pack the result before returning
                        ;15454 ;                               Temp2,Temp3 contain the new index The limit
                        ;15455 ;                               is validated and the new index written out
                        ;15456 ;                               If overflow exit is to FP.ENDUP1
                        ;15457 ;                               Otherwise the comparison as described in
                        ;15458 ;                               ABCF is made
                        ;15459 ;                               Note only if the 1st 32 are the same is it
                        ;15460 ;                               necessary to compare the low order 32 bits
                        ;15461 ;                               In either case these comparisons end up
                        ;15462 ;                               in common flows with the ACBF instruction.
                        ;15463 ;
                        ;15464 ;       **********NOT AN FPA INSTRUCTION**********
                        ;15465 ;
                        ;15466 ;*****************************************************************************
                        ;15467
                        ;15468 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;15469 FP.ACBD:                                ; **********************
                        ;15470         ;-------------------------------;
U 0359, 0086,95BE,4030,4047,0062,8      ;15471         M[TEMP9]_R[TEMP1]               ; SAVE BITS 0-31 OF LIMIT
                        ;15472
                        ;15473 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                        ;15474 =00
                        ;15475         ;00-------------------------------;
                        ;15476         R[TEMP8]_M[MDR].RR.16,          ; ROTATE LOW BITS 32-63
                        ;15477         LOD INC BRA?,                   ; RETURN IS THRU IRDROM TO FP.ACBD.100
U 0628, 0C85,23B7,0192,0047,0418,0      ;15478         PUSH,NEXT/OS.DRED               ; RETURN AFTER ADD ALL DONE
```

```
                                ;15479          ;01------------------------------;
                                ;15480          WB_EXP(M[TEMP9]),               ; TEST VALIDITY OF LIMIT
                                ;15481          FRO.FLTZ?,                      ;
U 0629, 0C80,9437,0AB0,0047,0C61,C 323*         ;15482          PUSH,NEXT/FP.ACBF.500           ;
                                ;15483
                                ;15484 FX.ACBG.345:
                                ;15485          ;10------------------------------; WRITE NEW INDEX
                                ;15486          R[DST.R] M[TEMP2],              ;
                                ;15487          WRITE NOTREG, SIZE[IDEP],       ;
                                ;15488          SET MM.NOINT,                   ; NO INTERRUPTS ALLOWED
U 062A, 0C64,2592,403C,45DA,046F,5              ;15489          PUSH,NEXT/FP.INCVA              ; BUMP UP VA
                                ;15490
                                ;15491 FX.ACBG.350:
                                ;15492          ;11------------------------------;
                                ;15493          R[DST.R+1] M[TEMP3],            ; WRITE SECOND HALF OF INDEX
                                ;15494          WRITE NOTREG,SIZE[LONG],        ; CONTINUE TESTING
                                ;15495          ADD1(FLAG0)?,                   ; DO NOT ENABLE INTERRUPTS
U 062B, 0084,3592,442F,45DA,006C,8              ;15496          NEXT/FP.ACBD.400                ;
                                ;15497
                                ;15498 =0
                                ;15499 FX.ACBD.400:
                                ;15500 FP.ACBD.400:
                                ;15501          ;0------------------------------; ADD.RD POSITIVE
                                ;15502          WB_Q-D,                         ; LIMIT -INDEX
                                ;15503          ALUS_SIGND,SIZE[LONG],          ; SAVE CONDITIONS
                                ;15504          WX.EQ.0?,                       ;
U 06C8, 0480,002B,0A20,18C7,086C,A 339*         ;15505          NEXT/FP.ACBD.450                ;
                                ;15506
                                ;15507          ;1------------------------------; ADD.RD NEGATIVE
                                ;15508          WB_D-Q,                         ; INDEX-LIMIT
                                ;15509          ALUS_SIGND,SIZE[LONG],          ; SAVE CONDITIONS
                                ;15510          WX.EQ.0?,                       ;
U 06C9, 0480,0028,0A20,18C7,086C,A 339*         ;15511          NEXT/FP.ACBD.450                ;
```

```
                                ;15512 =0
                                ;15513 FP.ACBD.450:
                                ;15514          ;0--------------------------------; FIRST PART NOT EQUAL
                                ;15515          ALUS?,                             ; WHAT WAS CONDITION
U 06CA, 0C80,0036,4B70,1847,0061,9 ;15516       NEXT/FP.ACBF.450                   ;
                                ;15517
                                ;15518          ;1--------------------------------; FIRST PARTS EQUAL MORE WORK
                                ;15519          D_M[TEMP3].RR.16 Q_R[TEMP8],       ; FIX UP SECOND PARTS
U 06CB, 0C80,33BD,7512,0047,0062,C ;15520       F[AG<1-0>?                         ; WERE BOTH OPERANDS NEGATIVE
                                ;15521
                                ;15522 =00
                                ;15523 FX.ACBH.500:                                ;
                                ;15524          ;00-------------------------------; BOTH POSITIVE ADDEND +
                                ;15525          WB_Q-D,                            ; LIMIT - INDEX
                                ;15526          ALUS_UNSGN,SIZE[LONG],             ; LATCH THE UNSIGNED COMPARE
U 062C, 0880,002B,0020,10C7,006C,A ;15527       NEXT7FP.ACBD.450                   ; FINISH UP
                                ;15528
                                ;15529          ;01-------------------------------; BOTH POSITIVE ADDEND -
                                ;15530          WB_D-Q,                            ; INDEX - LIMIT
                                ;15531          ALUS_UNSGN,SIZE[LONG],             ; LATCH THE UNSIGNED COMPARE
U 062D, 0880,0028,0020,10C7,006C,A ;15532       NEXT7FP.ACBD.450                   ; FINISH UP
                                ;15533
                                ;15534          ;10-------------------------------; BOTH NEGATIVE ADDEND +
                                ;15535          WB_D-Q,                            ; INDEX-LIMIT
                                ;15536          ALUS_UNSGN,SIZE[LONG],             ; LATCH THE UNSIGNED COMPARE
                                ;15537          CLEAR FLAG1,                       ; BE SURE FLAG CLEAR
U 062E, 0808,0028,0020,10C7,006C,A ;15538       NEXT/FP.ACBD.450                   ; FINISH UP
                                ;15539
                                ;15540          ;11-------------------------------; BOTH NEGATIVE ADDEND -
                                ;15541          WB_Q-D,                            ; LIMIT-INDEX
                                ;15542          ALUS_UNSGN,SIZE[LONG],             ; LATCH THE UNSIGNED COMPARE
                                ;15543          CLEAR FLAG1,                       ; BE SURE FLAG CLEAR
U 062F, 0808,002B,0020,10C7,006C,A ;15544       NEXT/FP.ACBD.450                   ; FINISH UP
                                ;15545
                                ;15546 FX.INCVA:
                                ;15547 FI.INCVA:
                                ;15548 FP.INCVA:
                                ;15549          ;--------------------------------;
                                ;15550          VA_VA+4,                           ; BUMP UP WRITE ADDRESS
U 06F5, 0080,0036,40B0,0447,0000,1 ;15551       RETURN [+1]                        ; RETURN TO CALLING ROUTINE
```

```
;15552 .TOC "  Floating point and CRC          : FP.POLYF "
;15553
;15554 ;*****************************************************************************
;15555 ;
;15556 ;       55      POLYF   arg.rf,degree.rw,tbladdr.ab [R0-R3.wl]
;15557 ;
;15558 ;               Input:
;15559 ;                       Q               argument (arg.rf)
;15560 ;                       MDR             degree (degree.rw)
;15561 ;               Resources:
;15562 ;                               for argument (FP.MULFOP1 to parse validate)
;15563 ;                       Temp2           fraction
;15564 ;                       Temp5           exponent
;15565 ;                       Flag2           sign of argument
;15566 ;                          (MUL1(FLAG2))
;15567 ;                               for Partial product (FP.MULFOP2  parse validate)
;15568 ;                       Temp0           fraction
;15569 ;                       RTemp8          exponent
;15570 ;                       Flag3           sign of operand
;15571 ;                          (MUL2(FLAG3))
;15572 ;                       Flag1           if operand 0
;15573 ;                          (MOPZERO(FLAG1))
;15574 ;                               for product step at end
;15575 ;                       Temp2           fraction
;15576 ;                       Temp5           exponent
;15577 ;                       Flag0           sign of product
;15578 ;                          (ADD1(FLAG0))
;15579 ;
;15580 ;                       for next coefficient (FP.MULFOP2 parse validate)
;15581 ;               Temp0           fraction
;15582 ;               Temp7           exponent
;15583 ;               Flag3           sign of coefficient after parse
;15584 ;                  (MUL2(FLAG3))
;15585 ;               Flag1           sign of coefficient for add operation
;15586 ;                  (ADD2(FLAG1))
;15587 ;
;15588 ;                       sum (i.e. partial product)
;15589 ;               R0              packed form
;15590 ;
;15591 ;                       General resources
;15592 ;               FPDOFFSET       MTemp containing FPD index value 1
;15593 ;               D,Q             multiplication and temp storage
;15594 ;
;15595 ;                       GPRs during instruction execution
;15596 ;               R0              1st coefficient or part product
;15597 ;               R1              argument (packed form)
;15598 ;               R2              byte 0 degree (decremented in situ)
;15599 ;                               byte 2 PC delta
;15600 ;                               byte 3 flag
;15601 ;                                       bit 30 read last coefficient
;15602 ;               R3              tbladdr
```

```
;15603 ;
;15604 ;        Output:
;15605 ;                R0              answer
;15606 ;                R1              0
;15607 ;                R2              0
;15608 ;                R3              tbladdr +degree*4+4
;15609 ;
;15610 ;        After fault detected if FPD = 1
;15611 ;
;15612 ;                R0              partial product after iteration prior
;15613 ;                                to the one causing fault
;15614 ;                R1              argument
;15615 ;                R2              <7:0> # of iterations remaining
;15616 ;                                <23-16> PC delta
;15617 ;                R3              address of coefficient
;15618 ;                                causing the exception
;15619 ;        Subroutines:
;15620 ;                FP.MULFOP1      parse operands
;15621 ;                FP.MULFOP2      parse operands
;15622 ;                FP.POLYFD.SUBS  initial set up
;15623 ;                IE.SERV.IP.TS2  check interrupt and timer
;15624 ;                FP.SC17         step counter 17
;15625 ;                FP.ADDRTN       do add operation
;15626 ;                FP.NORM.SNG     normalize,round,pack
;15627 ;        Operation:
;15628 ;                The argument is parsed and validated.
;15629 ;                FPDOFFSET temp is set to 1 for POLYF instruction
;15630 ;                FP.POLYFD.SUBS gets the  address of the coefficient
;15631 ;                table and does the initial preparations sets FPD
;15632 ;                Upon return the 1st coefficient is saved in R0 and the
;15633 ;                argument in R1
;15634 ;                If the argument is 0 then
;15635 ;                        if the degree is 0 the end up routine is called
;15636 ;                        if the degree ne 0 then the answer is the last
;15637 ;                                coefficient which is fetched
;15638 ;                                by adding degree*4 to the tbladdr
;15639 ;                                R2 bit 30 set to
;15640 ;                                indicate last coefficient
;15641 ;                                where to return should a fault occur
;15642 ;                                since FPD is already set
;15643 ;                If the argument is not 0
;15644 ;                        the main loop begins
;15645 ;                        if the interrupt or timer must be serviced
;15646 ;                        a call is made to IE.SERV.IP.TS2
;15647 ;                        if only a timer needs attention
;15648 ;                        return is -1 otherwise return is to the
;15649 ;                        FPD pack up routine
;15650 ;                        if the degree =>0 and no interrupt or timer
;15651 ;                        the part product is fetched from R0 and
;15652 ;                        unpacked the argument is also unpacked
;15653 ;                        the operands are shifted left 1 removing
;15654 ;                        the overflow position and the multiplication
;15655 ;                        performed resulting in a 32 bit product in D
```

```
                                                ;15656 ;                                31 bits of significance are saved in Temp2
                                                ;15657 ;                                and the operand unnormalized to bit 30
                                                ;15658 ;                                The next coefficient is read parsed (FP.MULFOP2)
                                                ;15659 ;                                The exponents are added and then properly biased
                                                ;15660 ;                                A call to FP.ADDRTN performs the add operation
                                                ;15661 ;                                FP.NORM.SNG normalizes,rounds,packs the product
                                                ;15662 ;                                if overflow or underflow(FU bit set) detected
                                                ;15663 ;                                a fault exit is taken
                                                ;15664 ;                                The degree count is decremented and a check
                                                ;15665 ;                                for interrupt or timer made
                                                ;15666 ;                                When all operands have been processed the
                                                ;15667 ;                                normal end up procedure is to  clear R1,R2
                                                ;15668 ;                                and to exit
                                                ;15669 ;
                                                ;15670 ;        **********FPA ENTRY AND INTERFACE**********
                                                ;15671 ;                (OS.FRED,OS.RED) - FI.POLYF
                                                ;15672 ;                FPD RESTART      - FI.POLYF.FPD.RES
                                                ;15673 ;
                                                ;15674 ;*************************************************************************
                                                ;15675 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                                ;15676 =000
                                                ;15677         ;000-----------------------------; RET -1 FROM FP.MULFOP1
                                                ;15678         R[TEMP2] D 0,                     ; ARG = 0
                                                ;15679         CLEAR MOPZERO(FLAG1),             ; FLAG NOT NEEDED MUST BE CLEAR FOR READ
U 01E0, 0C0C,05B7,2030,8047,001E,2              ;15680         NEXT/FP.POLYF.10                  ; CONTINUE PROCESSING FOR AWHILE
                                                ;15681
                                                ;15682 FP.POLYF:                                  ; ********************
                                                ;15683         ;001-----------------------------;
                                                ;15684         M[TEMP2] D 0  0 MB,               ; Q DESTROYED GET ARG TO MBUS SOURCE
U 01E1, 0C86,200D,7030,0047,0463,A              ;15685         PUSH,NEXT/FP.MULFOP1              ; PARSE TEST VALIDITY OF ARG
                                                ;15686
                                                ;15687 FP.POLYF.10:
                                                ;15688         ;010-----------------------------; RET +1 ARG OK
                                                ;15689         M[FPDOFFSET]_ZLITO[1],            ; SET FPDOFFSET REGISTER
U 01E2, 0186,0C37,0030,0847,0463,C              ;15690         PUSH,NEXT/FP.POLYFD.SUBS          ; AND SET FPD BEFORE RETURNING
                                                ;15691
                                                ;15692         ;011-----------------------------; HAVE ALL OPERANDS
                                                ;15693         R[R0]_M[MDR],                     ; SAVE FIRST COEFFICIENT FPD IS SET
                                                ;15694         CCOP1_SIZE[LONG],                 ; SET CONDITION CODES TENTATIVELY
U 01E3, 0E85,2592,4024,0847,046F,F              ;15695         PUSH,NEXT/FP.POLYFD.VAR3          ; SAVE VA IN R3
                                                ;15696
                                                ;15697         ;100-----------------------------; DEGREE MAY BE 0 IF SO 1ST COEFF
                                                ;15698         R[R1] D,WX.EQ.0?                  ; IS ANS. D HAS ARG
U 01E4, 0884,0582,4A34,4047,086D,2 322*         ;15699         NEXT/FP.POLYF.115                 ; CONTINUE ON
                                                ;15700 =
                                                ;15701 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
```

```
                                ;15702 =00
                                ;15703 FI.POLYF.20:
                                ;15704 FP.POLYF.20:
                                ;15705          ;00--------------------------------; DEG=0
                                ;15706          R[R3]_M[VA],CLEAR FPD,              ; R3 GET THE UPDATED ADDRESS
U 0630, 08E5,B592,4034,C047,006F,E ;15707       NEXT/FP.POLYFD.END                  ; FPD NOW CLEAR
                                ;15708
                                ;15709          ;01--------------------------------; DEG NE 0
                                ;15710          M[TEMP4]_MB-ZLIT0[4],               ; ACCOUNT FOR INCREMENTED VA
U 0631, 0186,4C10,0030,2047,0470,0 ;15711       PUSH,NEXT/FP.POLYFD.BT30            ; SET BIT 30 FOR SPECIAL READ
                                ;15712
                                ;15713          ;10--------------------------------;
U 0632, 0081,B001,0031,04A7,006F,6 ;15714       VA_M[VA]+R[TEMP4]                   ; INCREMENT VA TO POINT TO LAST COEF
                                ;15715 =
                                ;15716 FP.POLYF.22:
                                ;15717          ;--------------------------------;
                                ;15718          READ,SIZE[LONG],                    ; GET COEF
U 06F6, 0880,0036,4020,0450,006C,C ;15719       VA_VA+4                             ;
                                ;15720
                                ;15721 =0
                                ;15722 FP.POLYF.25:
                                ;15723          ;0---------------------------------;
                                ;15724          WB_EXP(M[MDR]),                     ; TEST VALIDITY OF LAST COEF
                                ;15725          FRO.FLTZ?,CLEAR FLAG1,              ;
U 06CC, 0C09,2437,0AB0,0047,0C5B,C 323* ;15726  PUSH,NEXT/FP.MULFOP2                 ;
                                ;15727
                                ;15728          ;1---------------------------------; HAVE LAST COEF
                                ;15729          R[R0]_M[MDR],                       ; SAVE IT IN R0
                                ;15730          CCOP1,SIZE[LONG],                   ; SET CONDITION CODES BASED ON IT
U 06CD, 0085,2592,4024,0847,0063,6 ;15731       NEXT/FP.POLYF.55                    ; FINISH UP INSTRUCTION
                                ;15732
                                ;15733 =00
                                ;15734 FP.POLYF.50:
                                ;15735          ;00--------------------------------; LOOP >=0, NO INT
                                ;15736          MDR_R[R0],CLEAR FLAG0,              ; GET PARTIAL PRODUCT
U 0634, 0000,05BE,4034,0467,006C,E ;15737       NEXT/FP.POLYF.60                    ; CONTINUE PROCESSING IN LOOP
                                ;15738
                                ;15739          ;01--------------------------------; LOOP >=0 INT OR TIMER
                                ;15740          D_ZLIT0[0],                         ; CLEAR OUT D FOR FPD FAULT
                                ;15741          INTPEND OR TIMER?,                  ;
U 0635, 0980,0C37,2AF0,0047,04F7,0 ;15742       PUSH,NEXT/IE.SERV.IP.TS2            ; RETURN -1 IF ONLY TIMER
                                ;15743
                                ;15744 FP.POLYF.55:
                                ;15745          ;10--------------------------------; LOOP <0 DONE
                                ;15746          R[R1]_0,                            ; START THE FINISH UP
U 0636, 0484,05B7,0034,4047,0063,0 ;15747       NEXT/FP.POLYF.20                    ; FINISH UP
                                ;15748 =
```

```
                                ;15749 =0
                                ;15750 FP.POLYF.60:
                                ;15751          ;0------------------------------;
                                ;15752          R[TEMP8]_EXP(M[MDR]),           ; UNPACK PARTIAL PRODUCT
                                ;15753          CLEAR MOPZERO(FLAG1),           ; BE SURE FLAG IS CLEAR
                                ;15754          FRO.FLTZ?,                      ; PARSE SIGN (MUL2(FLAG3))
U 06CE, 0C0D,2437,0AB2,0047,0C5B,C 323*  ;15755          PUSH,NEXT/FP.MULFOP2            ; RET TEMP0 (FRACT), TEMP8 (EXP)
                                ;15756
                                ;15757          ;1------------------------------;
                                ;15758          READ,SIZE[LONG],VA_VA+4,        ; GET COEF INTO MDR
                                ;15759          CLEAR MOPZERO(FLAG1),           ; BE SURE IT IS CLEAR AGAIN
U 06CF, 0008,0036,4020,0450,0054,A       ;15760          NEXT/FP.POLYF.62                ; SKIP AROUND RETURN
                                ;15761
                                ;15762 =000
                                ;15763 =001
                                ;15764          ;001-----------------------------; SHOULD NOT GET RET-1
                                ;15765          CLEAR MOPZERO(FLAG1),           ; BUT BULLET PROOF IT FOR FPD
U 0549, 0408,0036,4030,0047,0054,B       ;15766          NEXT/FP.POLYF.63                ; CONTINUE ON
                                ;15767
                                ;15768 FP.POLYF.62:
                                ;15769          ;010-----------------------------;
                                ;15770          M[TEMP2]_R[R1],                 ; UNPACK ARG CANT GET ZERO HERE
U 054A, 0886,25BE,4034,4047,0463,A       ;15771          PUSH,NEXT/FP.MULFOP1            ; TEMP2 & Q (FRACT) TEMP5 (EXP)
                                ;15772
                                ;15773 FP.POLYF.63:
                                ;15774          ;011-----------------------------;
                                ;15775          STEPC_ZLIT0[17.],               ; SET STEPC TO 17
                                ;15776          MUL1(FLAG2) XOR MUL2(FLAG3)?,   ; PRODUCT SIGN
U 054B, 0180,0C37,0570,8907,0455,6       ;15777          PUSH,NEXT/FP.POLYF.75           ;
                                ;15778
                                ;15779 FP.POLYF.65:
                                ;15780          ;100-----------------------------; THE MULTIPLY
                                ;15781          MULFAST+ CAND IN R[TEMP0],      ;
                                ;15782          DBZ STEPC?,SIZE[LONG],          ; DONE YET
U 054C, 0C80,0027,9320,0047,0054,C       ;15783          NEXT/FP.POLYF.65                ;
                                ;15784
                                ;15785          ;101-----------------------------;
                                ;15786          R[TEMP7]_EXP(M[MDR]),           ; PARSE NEXT COEFFICIENT
                                ;15787          CLEAR MUL2(FLAG3),              ; BE SURE FLAG IS CLEAR
                                ;15788          FRO.FLTZ?,                      ;
U 054D, 041D,2437,0AB1,C047,0C5B,C 323*  ;15789          PUSH,NEXT/FP.MULFOP2            ;
                                ;15790
                                ;15791          ;110-----------------------------; ALL DONE
                                ;15792          M[TEMP2]_D,                     ; UNNORMALIZE TO BIT 30 TEST
                                ;15793          WB<31-30>?,                     ; IS BIT 31 SET OR IS FRACT 0?
U 054E, 0086,25B2,46F0,0047,013F,4       ;15794          NEXT/FP.POLYF.FIX25.00          ;******FIX FOR VER 25
                                ;15795 =
```

```
                                ;15796 639:
                                ;15797 FREE.0639:                          ; NOT USED IN VERSION 25*******
                                ;15798         ;01-------------------------; VALUE IS 01XXXXXX......
                                ;15799         R[TEMP5].SIZ_RB-1,          ; MOVE BP DEC EXP
                                ;15800         SIZE[LONG],                 ;
U 0639, 0C82,0E7D,0021,4047,006F,7 ;15801      NEXT/FP.POLYF.68            ; KEEP ONLY 31 BITS OF SIGNIFICANCE
                                ;15802 63B:
                                ;15803 FREE.063B:                          ; NOT USED IN VERSION 25*****
                                ;15804         ;11-------------------------; VALUE IS 1XXXXXXX.....
                                ;15805         M[TEMP2]_MB.SR.1,           ; SHIFT IT RIGHT ONE
                                ;15806         MOPZERO(FLAG1)?,            ; WAS THE COEFFICIENT 0
U 063B, 0486,2001,85FD,8047,0060,0 ;15807      NEXT/FP.POLYF.70            ;
                                ;15808
                                ;15809 FP.POLYF.68:
                                ;15810         ;-------------------------;
                                ;15811         M[TEMP2]_MB.AND.OLIT0[1FE], ; GET RID OF 32ND BIT OF SIGNIFICANCE
U 06F7, 0586,2E12,05FF,F047,0060,0 ;15812      MOPZERO(FLAG1)?             ; WAS THE COEFFICIENT 0
                                ;15813 =0*
                                ;15814 FP.POLYF.70:
                                ;15815         ;0*-------------------------; COEFFICIENT NOT ZERO
                                ;15816         M[TEMP5]_MB+R[TEMP8],       ; ADD EXPONENTS
                                ;15817         SIZE[WORD],                 ; USE WORD SIZE FOR OVERFLOW
                                ;15818         MUL2(FLAG3)?,               ; CONVERT FLAG TO FLAG2
U 0600, 0886,5001,04D2,0047,006D,0 ;15819      NEXT/FP.POLYF.85            ;
                                ;15820
                                ;15821         ;1*-------------------------; COEFFICIENT IS ZERO
                                ;15822         M[TEMP5]_MB+R[TEMP8],       ; ADD EXPONENTS
                                ;15823         SIZE[WORD],                 ; USE WORD SIZE FOR OVERFLOW
U 0602, 0C16,5001,0012,0047,006F,8 ;15824      CLEAR MUL1(FLAG2)           ;
                                ;15825
                                ;15826         ;-------------------------;
                                ;15827         M[TEMP5]_MB-ZLIT0[80],      ; REMOVE THE BIAS
                                ;15828         SIZE[WORD],                 ;
                                ;15829         CLEAR MOPZERO(FLAG1),       ; CLEAR THE FLAG
U 06F8, 0D0E,5C10,0014,0047,0055,3 ;15830      NEXT/FP.POLYF.100           ;
                                ;15831
                                ;15832 =10
                                ;15833 FP.POLYF.75:
                                ;15834         ;10-------------------------; PRODUCT IS POSITIVE
                                ;15835         CLEAR ADD1(FLAG0),          ; CLEAR THE SIGN FLAG
                                ;15836         M[TEMP0]_(MB Q).SL.1,       ; SHIFT OPERANDS LEFT 1
U 0556, 0006,0205,C0BD,8047,0000,1 ;15837      RETURN [+1]                 ;
                                ;15838
                                ;15839         ;11-------------------------; PRODUCT IS NEGATIVE
                                ;15840         SET ADD1(FLAG0),            ; SET THE SIGN FLAG
                                ;15841         M[TEMP0]_(MB Q).SL.1,       ; SHIFT OPERANDS LEFT 1
U 0557, 0846,0205,C0BD,8047,0000,1 ;15842      RETURN [+1]                 ;
```

```
                                ;15843 13F4:
                                ;15844 FP.POLYF.FIX25.00:                       ; FIX FOR VER 25*****
                                ;15845          ;00-----------------------------; FRACTION IS 0
                                ;15846          M[TEMP5]_ZLIT0[80],             ; SET UP TO CLEAR EXP FIELD
U 13F4, 0186,5C37,0034,0047,013F,3 ;15847       NEXT/FP.POLYF.FIX25.10          ;
                                ;15848 13F5:
                                ;15849          ;01-----------------------------; VALUE IS 01XXXXXX......
                                ;15850          R[TEMP5].SIZ_RB-1,              ; MOVE BP DEC EXP
                                ;15851          SIZE[LONG],                     ;
U 13F5, 0C82,0E7D,0021,4047,006F,7 ;15852       NEXT/FP.POLYF.68                ; KEEP ONLY 31 BITS OF SIGNIFICANCE
                                ;15853 13F6:
                                ;15854          ;10-----------------------------; VALUE IS 1XXXXXXX.....
                                ;15855          M[TEMP2]_MB.SR.1,               ; SHIFT IT RIGHT ONE
                                ;15856          MOPZERO(FLAG1)?,                ; WAS THE COEFFICIENT 0
U 13F6, 0486,2001,85FD,8047,0060,0 ;15857       NEXT/FP.POLYF.70                ;
                                ;15858 13F7:
                                ;15859          ;11-----------------------------; VALUE IS 1XXXXXXX.....
                                ;15860          M[TEMP2]_MB.SR.1,               ; SHIFT IT RIGHT ONE
                                ;15861          MOPZERO(FLAG1)?,                ; WAS THE COEFFICIENT 0
U 13F7, 0486,2001,85FD,8047,0060,0 ;15862       NEXT/FP.POLYF.70                ;
                                ;15863
                                ;15864
                                ;15865 13F3:
                                ;15866 FP.POLYF.FIX25.10:
                                ;15867          ;-----------------------------;
                                ;15868          R[TEMP8]_0,                     ; ZERO OUT OTHER EXP FIELD
                                ;15869          MOPZERO(FLAG1)?,                ; SIGN OF NEXT COEF
U 13F3, 0084,05B7,05F2,0047,0060,0 ;15870       NEXT/FP.POLYF.70                ;
```

```
                                        ;15871 =0
                                        ;15872 FP.POLYF.85:
                                        ;15873          ;0------------------------------; COEF SIGN POSITIVE
                                        ;15874          M[TEMP5]_MB-ZLIT0[80],          ; REMOVE EXTRA BIAS FROM MULT STAGE
                                        ;15875          SIZE[WORD],CLEAR ADD2(FLAG1),   ; BE SURE FLAG IS CLEAR
U 06D0, 050E,5C10,0014,0047,0055,1      ;15876          NEXT/FP.POLYF.90                ; GO DO ADD ROUTINE
                                        ;15877
                                        ;15878          ;1------------------------------; COEF SIGN NEGATIVE
                                        ;15879          M[TEMP5]_MB-ZLIT0[80],          ; REMOVE EXTRA BIAS
U 06D1, 0D4E,5C10,0014,0047,0055,1      ;15880          SIZE[WORD],SET ADD2(FLAG1)       ; SET SIGN FLAG
                                        ;15881
                                        ;15882 =000
                                        ;15883 =001
                                        ;15884 FP.POLYF.90:
                                        ;15885          ;001-----------------------------;
                                        ;15886          PL_D_M[TEMP5]-R[TEMP7],         ; SUB EXPONENTS
                                        ;15887          CLEAR MUL1(FLAG2),              ; BE SURE FLAG CLEAR
                                        ;15888          SIGND CMP?,SIZE[WORD],          ; WHICH EXP BIGGER
U 0551, 0810,5BC0,2B51,C047,0C51,0 374* ;15889          PUSH,NEXT/FP.ADDRTN             ; GO DO THE ADD
                                        ;15890
                                        ;15891          ;010-----------------------------; RET +1 ANS IS SRC2 (COEF)
                                        ;15892          R[R0]_M[MDR],                   ; SAVE NEW PART PRODUCT
                                        ;15893          CCOP1,SIZE[LONG],               ; SET CONDITION CODE S ON THIS ROUND
U 0552, 0085,2592,4024,0847,006D,2      ;15894          NEXT/FP.POLYF.115               ; END UP THIS ROUND
                                        ;15895
                                        ;15896 FP.POLYF.100:
                                        ;15897          ;011-----------------------------; RET +2 ANS IS PART PRODUCT
                                        ;15898          CLEAR MUL2(FLAG3),              ; TEMP2 (FRACT) TEMP5(EXP)
                                        ;15899          ADD1(FLAG0)?,                   ; CONVERT FLAG TO D VALUE
U 0553, 0018,0036,4430,0047,046D,8      ;15900          PUSH,NEXT/FP.DSIGNSET           ; ADD1 FLAG SIGN OF ANS
                                        ;15901
                                        ;15902          ;100-----------------------------; RET +3  ANS IS IN TEMP2  FRACT)
                                        ;15903          M[TEMP5]_MB-ZLIT0[1E],          ; SUB FACTOR FOR NORMALIZ  S
                                        ;15904          CLEAR MUL2(FLAG3),              ; MUST CLEAR THIS FLAG ALS
U 0554, 091E,5C10,0030,F047,0468,C      ;15905          PUSH,NEXT/FP.NORM.SNG           ; TIME TO NORMALIZE
                                        ;15906
                                        ;15907 FP.POLYF.112:
                                        ;15908          ;101-----------------------------;
                                        ;15909          R[R0]_M[TEMP2],                 ; PUT PART PRODUCT PACKED IN REGISTER
                                        ;15910          CCOP1,SIZE[LONG],               ; SET CONDITION CODES
U 0555, 0C04,2592,4024,0847,006D,2      ;15911          CLEAR ADD1(FLAG0)               ; DONE WITH THIS FLAG
                                        ;15912 =
                                        ;15913 =0
                                        ;15914 FP.POLYF.115:
                                        ;15915          ;0------------------------------; ARG NE 0
                                        ;15916          R[R2].SIZ_RB-1,                 ; R2  DEGREE BYTE 0, PC DELTA BYTE 2
                                        ;15917          SIZE[BYTE],CLEAR FLAG1,         ;
                                        ;15918          COUNT OR INT TIMER?,            ; IS LOOP COMPLETE OR IS
U 06D2, 040A,0E7D,0B04,8047,0863,4 410* ;15919          NEXT/FP.POLYF.50                ; TIME FOR INTERRUPT OR TIMER
                                        ;15920
                                        ;15921          ;1------------------------------; ARG = 0
                                        ;15922          M[TEMP4]_(MB+R[TEMP4]).SL.1,    ; COMPUTE DEGREE * 4
U 06D3, 0886,4001,CA71,0047,0863,0 350* ;15923          WX.NE.0?,NEXT/FP.POLYF.20       ; WAS DEGREE ZERO
```

```
;15924 .TOC "  Floating point and CRC          : COMMON POLYF AND POLYD SUBROUTINES "
;15925
;15926 ;******************************************************************************
;15927 ;
;15928 ;       FP.POLYF.SUBS
;15929 ;
;15930 ;               Common flow Initialization routine for POLYF,POLYD
;15931 ;
;15932 ;               Input:
;15933 ;                       MDR             degree
;15934 ;                       PC
;15935 ;                       PCBACK
;15936 ;
;15937 ;               Resources:
;15938 ;                       Temp4           degree zero extended
;15939 ;                       MDR             VA, coefficient,
;15940 ;                       R2              PC delta,byte 0-1
;15941 ;                                       byte 2 degree
;15942 ;                                       byte 3 bit 30
;15943 ;                       FPD             set
;15944 ;                       VA              table address
;15945 ;                       Temp1           coefficient
;15946 ;                       Q               coefficient
;15947 ;
;15948 ;               Subroutines:
;15949 ;                       OS.ADD          get tbladdr.ab
;15950 ;                       FP.MULFOP1      parse coefficient single
;15951 ;                       FP.MULDOP1      parse coefficient double
;15952 ;
;15953 ;
;15954 ;       Operation:
;15955 ;               The address of the coefficient table is obtained
;15956 ;               by calling OS.ADD and the value stored in VA
;15957 ;               FP.POLYFD.RD is called internally and the first
;15958 ;               coefficient is read into MDR VA is incremented
;15959 ;               If POLYF
;15960 ;                       The coefficient is parsed and validated by
;15961 ;                       FP.MULFOP2
;15962 ;               *** return is to validate the degree
;15963 ;               if >31 a reserved operand fault is taken
;15964 ;               otherwise the FPD is set the PC delta is
;15965 ;               calculated and merged with the degree in R2 and
;15966 ;               return is made to the main calling routine
;15967 ;
;15968 ;               If POLYD
;15969 ;                       Bits 0-31 are stored in Temp1
;15970 ;               the read flag is set to indicate the first
;15971 ;               half of the read has been completed
;15972 ;               if no fault occurs the coefficient is parsed
;15973 ;               by FP.MULDOP1 and a return
;15974 ;               is made to the routine above ***
```

```
                                ;15975 ;                       Other routines include
                                ;15976 ;                       FP.POLYFD.BT30  (BT30A)
                                ;15977 ;                               this sets bit 30 in R2 to indicate
                                ;15978 ;                               last coefficient read for POLYF and POLYD
                                ;15979 ;
                                ;15980 ;*****************************************************************************
                                ;15981
                                ;15982 =00
                                ;15983 FP.POLYFD.SUBS:
                                ;15984          ;00-------------------------------;
                                ;15985          R[TEMP4]_ZEXT(M[MDR]),          ; ZERO EXTEND THE DEGREE
                                ;15986          SIZE[WORD],PUSH,                ; FROM WORD TO LONG
                                ;15987          LOD INC BRA?,                   ; GET TBLADDR
U 063C, 0C85,259E,4191,0047,0412,0  ;15988      NEXT/OS.ADD                     ; GUARANTEED TO BE + NUMBER
                                ;15989
                                ;15990          ;01-------------------------------;
                                ;15991          VA_M[MDR],                      ; PUT TBLADDR IN VA TO DO READ
U 063D, 0081,2002,403D,84A7,046F,8  ;15992      PUSH,NEXT/FP.POLYFD.RD          ;
                                ;15993
                                ;15994 FX.POLYG.SUB:
                                ;15995          ;10-------------------------------;
                                ;15996          WB_M[TEMP4].ANDNOT.ZLIT0[01F],  ; IS DEGREES <=31
U 063E, 0580,4C13,8A30,F847,006D,4  ;15997      WX.EQ.0?                        ;
                                ;15998 =
                                ;15999 =0
                                ;16000          ;0--------------------------------; DEGREE>31
U 06D4, 0C80,0036,4030,0047,00FF,8  ;16001      NEXT/IE.OPER.FAULT              ; RESERVED OPERAND
                                ;16002
                                ;16003 FI.PCDELTA:
                                ;16004          ;1--------------------------------; DEGREE VALID MAY BE ZERO
                                ;16005          R[R2]_M[PC],                    ; SAVE THE PC
U 06D5, 08ED,A592,4034,8047,006F,9  ;16006      SET FPD                         ; SET FIRST PART DONE
                                ;16007
                                ;16008 FP.PCDELTA:
                                ;16009          ;-------------------------------;
U 06F9, 0085,9003,0034,8047,006F,A  ;16010      R[R2]_RB-M[PCBACK]              ; GET DIFF PC-PCBACK
                                ;16011
                                ;16012          ;-------------------------------;
                                ;16013          R[R2]_M[TEMP4].OR.(RB.RR.SIZ),  ; ROT PC DELTA 2BYTES(16 BITS) RIGHT
                                ;16014          SIZE[WORD],                     ; MERGE WITH DEGREE
U 06FA, 0C84,4292,4094,8047,0000,1  ;16015      RETURN [+1]                     ;
                                ;16016
                                ;16017 FP.POLYFD.RD:
                                ;16018          ;-------------------------------;
                                ;16019          READ,SIZE[LONG],                ; READ COEFF INTO MDR
                                ;16020          VA_VA+4,                        ; INCREMENT VA TO NEXT OPERAND
U 06FB, 0480,0036,48E0,0450,006D,6  ;16021      IR<5>?                          ; WAS IT POLYF OR POLYD
                                ;16022
                                ;16023 =0
                                ;16024          ;0--------------------------------; POLYF
                                ;16025          R[TEMP7]_EXP(M[MDR]),            ; PARSE AND TEST VALIDITY OF COEF
                                ;16026          FRO.FLTZ?,                      ;
U 06D6, 0485,2437,0AB1,C047,085B,C 323*  ;16027 NEXT/FP.MULFOP2                 ; RETURN IS TO CALLING ROUTINE
```

```
                                        ;16028          ;1-------------------------------; POLYD
                                        ;16029          R[TEMP1].SIZ_Q_M[MDR],          ; BITS 0-31 INTO TEMP1 AND Q
U 06D7, 084B,2592,5020,4047,006F,C      ;16030          SIZE[LONG],SET READ(FLAG1)      ;
                                        ;16031
                                        ;16032          ;-------------------------------;
                                        ;16033          READ,SIZE[LONG],                ; READ INTO MDR
                                        ;16034          CLEAR READ(FLAG1),              ;
U 06FC, 0008,0036,4020,0450,006F,D      ;16035          VA_VA+4                         ; BUMP UP THE ADDRESS
                                        ;16036
                                        ;16037 FP.POLYFD.25:
                                        ;16038          ;-------------------------------; CALLING FROM SPECIAL CASES
                                        ;16039          R[TEMP5]_EXP(M[TEMP1]),         ; PARSE COEF AND VALIDATE
                                        ;16040          FRO.FLTZ?,                      ; TEMP1, MDR PACKED VERSION
U 06FD, 0C84,1437,0AB1,4047,085E,4 323* ;16041          NEXT/FP.MULDOP1                 ; TEMP3 HAS BITS 32-63 FRACTION
                                        ;16042                                          ; Q HAS BITS 0-31 FRACT
                                        ;16043                                          ; MUL1(FLAG2) FOR SIGN AND TEMP5 FOR EXP
                                        ;16044
                                        ;16045 FP.POLYFD.END:
                                        ;16046          ;-------------------------------;
U 06FE, 0084,05B7,0134,8047,003F,9      ;16047          R[R2]_0,IRD1                    ; CLEAR REGISTER
                                        ;16048
                                        ;16049 FX.POLYG.VAR3:
                                        ;16050 FP.POLYFD.VAR3:
                                        ;16051          ;-------------------------------;
                                        ;16052          R[R3]_M[VA],                    ; SAVE THE VA
U 06FF, 0C85,B592,40B4,C047,0000,1      ;16053          RETURN [+1]                     ;
                                        ;16054
                                        ;16055 =0
                                        ;16056 FX.DSIGNSET:
                                        ;16057 FP.DSIGNSET:
                                        ;16058          ;0-------------------------------;
                                        ;16059          D_ZLIT0[0],                     ;
U 06D8, 0580,0C37,20B0,0047,0000,1      ;16060          RETURN [+1]                     ;
                                        ;16061
                                        ;16062          ;1-------------------------------;
                                        ;16063          D_ZLIT12[8],                    ;
U 06D9, 0180,0D77,20B0,4047,0000,1      ;16064          RETURN [+1]                     ;
                                        ;16065
                                        ;16066 FX.POLYFD.BT30:
                                        ;16067 FP.POLYFD.BT30:
                                        ;16068          ;-------------------------------;
U 0700, 0D80,0CB7,0032,0467,0070,1      ;16069          MDR_ZLIT24[40]                  ; SET UP FLAG BIT FOR SPECIAL READ
                                        ;16070
                                        ;16071 FP.POLYFD.BT30A:
                                        ;16072          ;-------------------------------;
                                        ;16073          R[R2].SIZ_M[MDR].OR.RB,         ; MERGE INTO CONTROL WORD
U 0701, 0C83,2002,40A4,8047,0000,1      ;16074          SIZE[LONG],RETURN [+1]          ;
```

```
                        ;16075 .TOC "  Floating point and CRC          : POLYF AND POLYD FIRST PART DONE SAVE "
                        ;16076
                        ;16077 ;*****************************************************************************
                        ;16078 ;
                        ;16079 ;       POLYF AND POLYD FIRST PART DONE ROUTINES
                        ;16080 ;
                        ;16081 ;               Pack up routines
                        ;16082 ;                       Initial instruction at FPDOFFSET #1 for POLYF
                        ;16083 ;                                              at FPDOFFSET #2 for POLYD
                        ;16084 ;
                        ;16085 ;                       There in both cases the iteration count is incremented
                        ;16086 ;                       by one and in the case of POLYD a check made to
                        ;16087 ;                       see if the fault had occurred during a read in which
                        ;16088 ;                       case the VA is decremented by 4.
                        ;16089 ;                       An additional check is made for a reserved operand fault
                        ;16090 ;                       If found the VA is decremented by 4/8 instead of the
                        ;16091 ;                       above and is saved in R3 and exit is to IE.PACK.DONE
                        ;16092 ;
                        ;16093 ;*****************************************************************************
                        ;16094 ;
                        ;16095 ;                       THIS IS CODE THAT APPEARS IN IANDE
                        ;16096 ;                                       IN FPDOFFSET #1 FOR  POLYF
                        ;16097 ;                                       IN FPDOFFSET #2 FOR  POLYD
                        ;16098 ;
                        ;16099 ;FP.POLYF.FPD.SAVE:
                        ;16100 ;       ;-------------------------------;
                        ;16101 ;       R[R2].SIZ_RB+1,SIZE[BYTE],      ; INCREMENT # OF ITERATIONS
                        ;16102 ;       NEXT/FP.POLYF.FPD.SAV.10        ;
                        ;16103 ;
                        ;16104 ;FP.POLYD.FPD.SAVE:
                        ;16105 ;       ;-------------------------------;
                        ;16106 ;       R[R2].SIZ_RB+1,SIZE[BYTE],      ; INCREMENT # OF ITERATIONS
                        ;16107 ;       READ(FLAG1)?,                   ; WAS THIS A READ SECOND HALF
                        ;16108 ;       NEXT/FP.POLYD.FPD.SAV.10        ;
                        ;16109
                        ;16110
                        ;16111 FP.POLYF.FPD.SAV.10:
                        ;16112         ;-------------------------------;
                        ;16113         Q_ZLIT0[4],                     ; SET Q WITH CONSTANT VALUE
U 0702, 0980,0037,1030,2047,0070,3      ;16114         NEXT/FP.POLYFD.FPD.10           ; FINISH UP RES OP CHECK
                        ;16115
                        ;16116 FP.POLYFD.FPD.10:
                        ;16117         ;-------------------------------;
                        ;16118         WB_D-ZLIT0[18],                 ; IS IT THE RES OP FAULT
                        ;16119         SIZE[BYTE],SIGND CMP DEF?,      ;
U 0703, 0980,0030,0840,1047,0850,2 413* ;16120         NEXT/FP.POLYFD.FPD.50           ; FINISH UP
```

```
                                ;16121 =10
                                ;16122 FP.POLYFD.FPD.50:
                                ;16123          ;10-------------------------------; NE 0
                                ;16124          R[R3] M[VA],                      ; SAVE VA
U 05C2, 0085,B592,4034,C047,00FE,2 ;16125       NEXT/IE.PACK.DONE                 ; FINISHED
                                ;16126
                                ;16127          ;11-------------------------------; EQ 0 RES OP
                                ;16128          R[R3] M[VA]-Q,                    ; DECREMENT VA
U 05C3, 0485,B008,0034,C047,00FE,2 ;16129       NEXT/IE.PACK.DONE                 ; FINISHED
                                ;16130
                                ;16131
                                ;16132 =0*
                                ;16133 FX.POLYG.FPD.SAV.10:
                                ;16134 FP.POLYD.FPD.SAV.10:
                                ;16135          ;0*-------------------------------; NOT SECOND HALF READ
                                ;16136          Q ZLIT0[8],                       ; SET UP FOR VA DECREMENT IF RES OP
U 0618, 0580,0C37,1030,4047,0070,3 ;16137       NEXT/FP.POLYFD.FPD.10             ;
                                ;16138
                                ;16139          ;1*-------------------------------; SECOND HALF READ
                                ;16140          VA M[VA]-ZLIT0[4],                ; BACK UP TO FIRST HALF OF DBLE READ
U 061A, 0D81,BC10,0030,24A7,005C,2 ;16141       NEXT/FP.POLYFD.FPD.50             ; NOW SAVE IT
```

```
;16142 .TOC "  Floating point and CRC          : POLYF POLYD FIRST PART DONE RESTART "
;16143
;16144 ;*****************************************************************************
;16145 ;
;16146 ;               POLYF AND POLYD FIRST PART DONE RESTART
;16147 ;
;16148 ;                       FP.POLYF.FPD.RES
;16149 ;
;16150 ;               Input:
;16151 ;                               R0              packed partial product
;16152 ;                               R1              packed argument
;16153 ;                               R2              byte 0-1 degree
;16154 ;                                               byte 2 PC delta
;16155 ;                                               byte 3 bit 30
;16156 ;                                               read last coefficient
;16157 ;                               R3              tbladdr
;16158 ;
;16159 ;               Resources:
;16160 ;                               FPDOFFSET       MTemp value #1 restored
;16161 ;                               VA
;16162 ;                               PC
;16163 ;                               Temp0
;16164 ;
;16165 ;               Operation:
;16166 ;                               Restore FPDOFFSET temp restore VA
;16167 ;                               recompute PC from PCBACK and PCdelta
;16168 ;                               if not read last coefficient
;16169 ;                               then decrement iteration count
;16170 ;                               and continue at FP.POLYF.50
;16171 ;                               otherwise read again and continue at
;16172 ;                               FP.POLYF.25 if successful
;16173 ;
;16174 ;                       FP.POLYD.FPD.RES
;16175 ;
;16176 ;               Input:
;16177 ;                               R0,R1           part product packed
;16178 ;                               R2              as for POLYF
;16179 ;                               R3              tbladdr
;16180 ;                               R4,R5           argument packed
;16181 ;
;16182 ;               Operation:
;16183 ;                               Get part product and unpack, get argument
;16184 ;                               and unpack and move to working temp
;16185 ;                               restore FPDOFFSET and VA
;16186 ;                               recompute PC from PCBACK and PC delta
;16187 ;                               if not read last coefficient
;16188 ;                               decrement iteration count and continue
;16189 ;                               at FP.POLYD.20
;16190 ;                               otherwise call FP.POLYD.35 to try again
;16191 ;                               return to do final end up if read is done
;16192 ;
;16193 ;*****************************************************************************
```

```
                                ;16194 .REGION/IRD1.R1L,IRD1.R1H
                                ;16195 =000
                                ;16196 FP.POLYF.FPD.RES:
                                ;16197          ;000-------------------------------;
                                ;16198          M[FPDOFFSET]_ZLIT0[1],          ; RESTORE FPDOFFSET
                                ;16199          PUSH,                           ; DO SOME COMMON RESTORATION
U 0200, 0986,CC37,0030,0847,0455,D ;16200       NEXT/FP.POLYFD.RES.10           ;
                                ;16201
                                ;16202          ;001------------------------------;
                                ;16203          WB_R[R2],                       ; DETERMINE WHERE TO RETURN
                                ;16204          WB<31-30>?,                     ; TESTING BIT30
U 0201, 0480,05BE,46F4,8047,0064,2 ;16205       NEXT/FP.POLYF.FPD.05            ;
                                ;16206 =
                                ;16207 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;16208 =10
                                ;16209 FP.POLYF.FPD.05:
                                ;16210          ;10-------------------------------;
                                ;16211          R[R2].SIZ_RB-1,                 ; DECREMENT THE # OF ITERATIONS
                                ;16212          SIZE[BYTE].CLEAR FLAG1,         ;
                                ;16213          COUNT OR INT TIMER?,            ; CHECK FOR INTERRUPTS TIMER
U 0642, 040A,0E7D,0B04,8047,0863,4 410* ;16214  NEXT/FP.POLYF.50                 ; CONTINUE THE PROCESSING
                                ;16215
                                ;16216          ;11-------------------------------;
                                ;16217          READ,SIZE[LONG],                ; DO SPECIAL READ AGAIN
                                ;16218          VA_VA+4,                        ;
U 0643, 0880,0036,4020,0450,006C,C ;16219       NEXT/FP.POLYF.25                ; JOIN REGULAR SPECIAL FLOW
                                ;16220 =
                                ;16221 .REGION/IRD1.R1L,IRD1.R1H
                                ;16222 =000
                                ;16223 FP.POLYD.FPD.RES:
                                ;16224          ;000------------------------------;
U 0220, 0486,15BE,4034,0047,0070,4 ;16225       M[TEMP1]_R[R0]                  ; GET PART PRODUCT UNPACKED
                                ;16226 =
                                ;16227 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;16228
                                ;16229          ;-------------------------------; SET UP FPDOFFSET FIRST
U 0704, 098C,CC37,0030,1047,0064,4 ;16230       M[FPDOFFSET]_ZLIT0[2]           ; FOR BULLET PROOFING
                                ;16231
                                ;16232 =00
                                ;16233          ;00-------------------------------;
                                ;16234          MDR_R[R1],                      ; SECOND HALF OF PART PRODUCT
U 0644, 0480,05BF,4034,4467,046F,D ;16235       PUSH,NEXT/FP.POLYFD.25          ; THIS UNPACKS  AND RETURNS
                                ;16236
                                ;16237          ;01-------------------------------;
                                ;16238          M[TEMP1]_R[R4],PUSH,            ; GET ARG AND UNPACK
U 0645, 0086,15BE,4035,0047,0457,3 ;16239       NEXT/FP.MT2Q                    ; GET Q (BITS 0-31) TO TEMP2
                                ;16240
                                ;16241          ;10-------------------------------;
U 0646, 0008,05BE,4035,4467,0064,8 ;16242       MDR_R[R5],CLEAR FLAG1           ; SECOND PART OF ARGUMENT
                                ;16243 =
```

```
                                ;16244 =00
                                ;16245          ;00------------------------------;
                                ;16246          R[TEMP8]_EXP(M[TEMP1]),         ; GO PARSE ARG
                                ;16247          FRO.FLTZ?,                      ;
0 0648, 0084,1437,0A82,0047,0C5E,8 323*         ;16248          PUSH,NEXT/FP.MULDOP2            ;
                                ;16249
                                ;16250          ;01------------------------------; HERE ONLY IF ZERO SHOULD NOT HAPPEN
0 0649, 0C80,0036,4030,0047,0064,A              ;16251          NEXT/FP.POLYFD.RES.08           ; FORCE TO FOLLOW THROUGH
                                ;16252
                                ;16253 FP.POLYFD.RES.08:
                                ;16254          ;10------------------------------; RET +2 FROM PARSE
                                ;16255          M[TEMP9]_R[TEMP1],              ; MOVE TO WORKING TEMP
0 064A, 0086,95BE,4030,4047,0455,D              ;16256          PUSH,NEXT/FP.POLYFD.RES.10      ;
                                ;16257
                                ;16258          ;11------------------------------;
0 064B, 0C86,45BE,4031,8047,0070,5              ;16259          M[TEMP10]_R[TEMP6]              ; MOVE TO WORKING TEMP
                                ;16260
                                ;16261          ;------------------------------;
                                ;16262          WB_R[R2],                       ; WHERE TO RETURN TO
0 0705, 0480,05BE,46F4,8047,0055,A              ;16263          WB<31-30>?                      ; BIT 30 TEST
                                ;16264 =000
                                ;16265 =010
                                ;16266          ;010-----------------------------; NORMAL PROCESSING
                                ;16267          R[R2].SIZ_RB-1,                 ; DECREMENT THE ITERATION COUNT
                                ;16268          SIZE[BYTE],                     ;
                                ;16269          SIGND CMP DEF?,                 ; DONE OR NOT
0 055A, 0882,0E7D,0B44,8047,0865,5 407*         ;16270          NEXT/FP.POLYD.20                ; CONTINUE ON
                                ;16271
                                ;16272          ;011-----------------------------; SPECIAL CASES WORK
                                ;16273          R[TEMP4]_0,                     ;
0 055B, 0084,05B7,0031,0047,0456,D              ;16274          PUSH,NEXT/FP.POLYD.35           ;
                                ;16275
                                ;16276          ;100-----------------------------;
                                ;16277          R[R3]_M[VA],CLEAR FPD,          ;
0 055C, 08E5,B592,4034,C047,006F,E              ;16278          NEXT/FP.POLYFD.END              ;
                                ;16279
                                ;16280 FI.POLYFD.RES.10:
                                ;16281 FP.POLYFD.RES.10:
                                ;16282 FX.POLYFD.RES.10:
                                ;16283          ;------------------------------;
0 055D, 0880,056F,4034,C4A7,0055,F              ;16284          VA_R[R3]                        ; RESTORE THE TBLADDR
                                ;16285
                                ;16286          ;------------------------------;
0 055E, 0086,02B7,0014,8047,0055,F              ;16287          M[TEMP0]_R[R2].RR.16            ; GET THE SAVED WORD AND FIND PCDELTA
                                ;16288
                                ;16289          ;------------------------------;
0 055F, 0D86,0C12,0037,F847,0070,6              ;16290          M[TEMP0]_MB.AND.ZLITO[OFF]      ; MASK OFF PCDELTA
                                ;16291
                                ;16292          ;------------------------------;
                                ;16293          PC_M[PCBACK]+R[TEMP0],          ; RESTORE PC CORRECTLY
0 0706, 0481,9001,0080,0487,0000,1              ;16294          RETURN [+1]                     ;
```

```
;16295 .TOC "  Floating point and CRC          : FP.POLYD "
;16296
;16297 ;*****************************************************************************
;16298 ;
;16299 ;       75      POLYD   arg.rd,degree.rw,tbladdr.ab,[R0-R5.wl]
;16300 ;
;16301 ;               Input:
;16302 ;
;16303 ;                       Temp1           argument bits 0-31
;16304 ;                       MDR             argument bits 32-63
;16305 ;
;16306 ;               Resources:
;16307 ;
;16308 ;                               for argument (FP.MULDOP2 parse validate)
;16309 ;                       Temp1,MTemp9    fraction bits 0-31
;16310 ;                       Temp6,MTemp10   fraction bits 32-63
;16311 ;                       RTemp8          exponent
;16312 ;                       Flag3           sign of operand
;16313 ;                          (MUL2(FLAG3))
;16314 ;                       MTemp8,MDR      packed form of operand initially
;16315 ;                       MTemp10,Q       packed form of operand initially only
;16316 ;
;16317 ;                               for partial product
;16318 ;                       Temp2           fraction bits 0-31
;16319 ;                       Temp3           fraction bits 32-63
;16320 ;                       Temp5           exponent
;16321 ;                       Flag2   sign of operand
;16322 ;                          (MUL1(FLAG2))
;16323 ;
;16324 ;                               for product step before getting coefficient
;16325 ;                       Temp2           fraction bits 0-31
;16326 ;                       Temp3           fraction bits 32-63
;16327 ;                       Temp5           exponent
;16328 ;                       Flag0           sign of product
;16329 ;                          (ADD1(FLAG0))
;16330 ;
;16331 ;                               General resources
;16332 ;                       FPDOFFSET       MTemp containing FPD index value 2
;16333 ;                       Flag1           special read flag
;16334 ;                          (READ(FLAG1))
;16335 ;                       D,Q             multiplication and temp storage
;16336 ;
;16337 ;                               GPRs during instruction execution
;16338 ;                       R0,R1           1st coefficient or part product
;16339 ;                       R2              byte 0 degree (decremented in situ)
;16340 ;                                       byte 2 PC delta
;16341 ;                                       byte 3 flag
;16342 ;                                               bit 30 special read
;16343 ;                       R3              tbladdr
;16344 ;                       R4,R5           argument in packed form
```

```
;16345 ;
;16346 ;        Output:
;16347 ;                R0,R1           answer
;16348 ;                R2              0
;16349 ;                R3              tbladdr + degree*8+8
;16350 ;                R4,R5           0
;16351 ;
;16352 ;        After fault if FPD = 1
;16353 ;
;16354 ;                R0,R1           partial product after the iteration
;16355 ;                                prior to the one causing the fault
;16356 ;                R2              <7:0>  # of iterations remaining
;16357 ;                                <23:16> PC delta
;16358 ;                R3              address of the coefficient
;16359 ;                                causing the fault
;16360 ;                R4,R5           argument
;16361 ;
;16362 ;        Subroutines:
;16363 ;                FP.MULDOP1      parse operands
;16364 ;                FP.MULDOP2      parse operands
;16365 ;                OS.RED          get degree
;16366 ;                FP.POLYFD.SUBS  initial set up
;16367 ;                FP.POLYFD.RD    do special read
;16368 ;                FP.PREOPS.POLYD prepare operands call FP.MULD.SUB
;16369 ;                FP.POLYFD.RD10  get next coefficient call FP.ADDD.20
;16370 ;                                to add, normalize,round pack
;16371 ;                IE.SERV.IP.TS2  process interrupt or timer
;16372 ;
;16373 ;        Operation:
;16374 ;
;16375 ;                The argument is parsed and validated by FP.MULDOP2
;16376 ;                The degree is fetched by OS.RED. Upon return
;16377 ;                FPDOFFSET temp is set to 2 for POLYD instruction
;16378 ;                FP.POLYFD.SUBS gets the  address of the coefficient
;16379 ;                table and does the initial preparations sets FPD
;16380 ;                Upon return the 1st coefficient is saved in R0,R1
;16381 ;                the argument in R3,R4
;16382 ;                If the argument is 0 then
;16383 ;                        if the degree is 0 the end up routine is called
;16384 ;                        if the degree ne 0 then the answer is the last
;16385 ;                                coefficient which is fetched
;16386 ;                                by adding degree*8 to the tbladdr
;16387 ;                                R2 (bit 30) is set indicating
;16388 ;                                where to return should a fault occur
;                                       since FPD is already set
```

```
;16389 ;                        If the argument is not 0
;16390 ;                                The exponents are added and FP.PREOPS.POLYD
;16391 ;                                called to shift the operands
;16392 ;                                left and to call FP.MULD.SUB to perform the
;16393 ;                                multiplication.  Upon return 63 bits of the
;16394 ;                                product are saved and the next
;16395 ;                                coefficient is obtained and FP.ADDD.20 called
;16396 ;                                to perform the add, normalize round pack
;16397 ;                                Upon return the part product is saved in R0,R1
;16398 ;                                if no floating overflow or underflow FU bit set
;16399 ;                                At appropriate intervals checks are also
;16400 ;                                made for interrupts and timer .
;16401 ;                                To begin a second or nth iteration
;16402 ;                                the part product is unpacked and
;16403 ;                                the operation repeated until all coefficients
;16404 ;                                have been processed
;16405 ;
;16406 ;          **********FPA ENTRY AND INTERFACE**********
;16407 ;                  (OS.FIDRED)     - FI.POLYD
;16408 ;                  FPD RESTART     - FI.POLYD.FPD.RES
;16409 ;
;16410 ;*******************************************************************************
```

```
                                ;16411 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;16412 FP.POLYD:                          ; **********************
                                ;16413          ;-------------------------------;
U 035A, 0C86,85BE,4030,4047,0064,C ;16414       M[TEMP8]_R[TEMP1]         ; SAVE BITS 0-31 OF ARG IN PACKED FORM
                                ;16415
                                ;16416 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;16417 =00
                                ;16418          ;00-------------------------------;
                                ;16419          R[TEMP8]_EXP(M[TEMP1]),   ; PARSE ARG TEMP1 BITS0-31 (FRACT)
                                ;16420          PUSH,                     ; MDR PACKED BITS 32-63
                                ;16421          FRO.FLTZ?,                ; TEMP6 HAS UNPACKED BITS32-63
U 064C, 0084,1437,0AB2,0047,0C5E,8 323* ;16422  NEXT/FP.MULDOP2            ;
                                ;16423
                                ;16424          ;01-------------------------------; RET +1 ARG = 0
U 064D, 0086,85BE,403D,8047,0064,E ;16425       M[TEMP8]_R[ZERO]          ; ZERO OUT PACKED ARG (BITS0-31)
                                ;16426
                                ;16427          ;10-------------------------------; RET +2 ARG NE 0
                                ;16428          M[TEMP9]_R[TEMP1],        ; SAVE THE FRACTION BITS 0-31
                                ;16429          LOD INC BRA?,             ; GO GET DEGREE
U 064E, 0886,95BE,4180,4047,0010,0 ;16430       NEXT/OS.REV               ; RET THRU IRDROM TO FP.POLYD.10
                                ;16431 =
                                ;16432 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;16433 FP.POLYD.10:
                                ;16434          ;-------------------------------;
U 035B, 0986,CC37,0030,1047,0065,0 ;16435       M[FPDOFFSET]_ZLIT0[2]     ; SET UP THE FPD OFFSET VALUE FOR POLYD
                                ;16436
                                ;16437 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;16438 =00
                                ;16439          ;00-------------------------------;
                                ;16440          M[TEMP10]_Q,              ; THE PACKED ARG(BITS32-63) MDR MOVED
                                ;16441          PUSH,                     ; TO Q BY OS.RED GET TBLADDR,SET FPD,
U 0650, 0C86,A03A,403D,8047,0463,C ;16442       NEXT/FP.POLYFD.SUBS       ; AND GET 1ST COEF
                                ;16443                                    ; TEMP2,TEMP3,TEMP5(EXP), UNPACKED
                                ;16444                                    ; TEMP1 AND MDR HAVE THE PACKED FORM
                                ;16445
                                ;16446          ;01-------------------------------;
                                ;16447          M[TEMP2]_Q,               ; UNPACKED BITS 0-31 INTO WORKING TEMP
                                ;16448          PUSH,                     ; 1ST COEFF (PART PRODUCT)
U 0651, 0086,203A,403D,8047,0456,E ;16449       NEXT/FP.POLYD.40          ; PUT PACKED VALUE IN R0,R1
                                ;16450
                                ;16451          ;10-------------------------------;
                                ;16452          R[R5]_M[TEMP10],          ; PACKED BITS 32-63 OF ARG (OR 0) IN REG
U 0652, 0884,A592,4035,4047,046F,F ;16453       PUSH,NEXT/FP.POLYFD.VAR3  ; GO SET R3 EQUAL TO VA
                                ;16454
                                ;16455          ;11-------------------------------;
U 0653, 0486,A5BE,4031,8047,002A,1 ;16456       M[TEMP10]_R[TEMP6]        ; SAVE BITS 32-63 OF UNPACKED ARG
                                ;16457
                                ;16458 02A1:
                                ;16459          ;*******FORCE ADDRESS***********;
                                ;16460          R[R4]_M[TEMP8],CLEAR FLAG1, ; STORE BITS 0-31 (OR 0) OF ARG
                                ;16461          WX.EQ.0?,                 ; WAS ARG 0
U 02A1, 000C,8592,4A35,0047,0856,8 322* ;16462  NEXT/FP.POLYD.25            ;
```

```
                                    ;16463 0707:
                                    ;16464 FREE.0707:
                                    ;16465          ;******LOCATION NOT USED *******;
                                    ;16466          R[R4]_M[TEMP8],                 ; STORE BITS 0-31 (OR 0) OF ARG
                                    ;16467          WX.EQ.0?,                       ; WAS ARG 0
U 0707, 0084,8592,4A35,0047,0856,8 322* ;16468      NEXT/FP.POLYD.25                ;
                                    ;16469 =01
                                    ;16470 FP.POLYD.20:
                                    ;16471          ;01-----------------------------; LOOP >=0
                                    ;16472          M[TEMP5]_MB+R[TEMP8],           ; ADD EXPS IN PREP FOR MULD RTN
                                    ;16473          CLEAR READ(FLAG1),              ; DOING A READ NORMALLY
U 0655, 040E,5001,0012,0047,0056,0  ;16474          SIZE[WORD],NEXT/FP.POLYD.22     ; ADDITION IN WORD SIZE
                                    ;16475
                                    ;16476          ;11-----------------------------; LOOP <0 ALL DONE
                                    ;16477          R[R4]_0,                        ; ZERO OUT WORKING REGS
U 0657, 0C84,05B7,0035,0047,0070,F  ;16478          NEXT/FP.POLYD.220               ;
                                    ;16479 =000
                                    ;16480 FP.POLYD.22:
                                    ;16481          ;000----------------------------;
                                    ;16482          M[TEMP5]_MB-ZLIT0[80],          ; REMOVE EXTRA BIAS
U 0560, 0186,5C10,0034,0047,046A,6  ;16483          PUSH,NEXT/FP.SC17
                                    ;16484
                                    ;16485          ;001----------------------------;
                                    ;16486          WB_R[R4].RR.SIZ,                ; CHECK AGAIN FOR ARG SIGN
                                    ;16487          SIZE[WORD],WB<31-30>?,          ; PREPARE OPS FOR MULTIPLY
U 0561, 0080,02B7,06D5,0047,0C5E,D 337* ;16488      PUSH,NEXT/FP.PREOPS.POLYD       ; AND MULTIPLY
                                    ;16489
                                    ;16490          ;010----------------------------;
                                    ;16491          M[TEMP2]_D+R[TEMP0]+ALKC,        ; ADD THE LAST PART OF CROSS PRODUCT
                                    ;16492          WB<31-30>?,                     ; UNNORMALIZE RET SOONER OR LATER
U 0562, 0086,2061,06F0,0047,0D3F,8 338* ;16493      PUSH,NEXT/FP.POLYD70.FIX        ; FIX IN VER 25******
                                    ;16494
                                    ;16495          ;011----------------------------; RET AFTER READ,ADD (AFTER FP.ADDD.20)
                                    ;16496          R[R0]_M[TEMP2],CCOP1,           ; BITS 0-31 IN MTEMP2
U 0563, 0C84,2592,4024,0847,006D,A  ;16497          SIZE[LONG]                      ; BITS 32-63 IN MTEMP3
                                    ;16498 =
                                    ;16499 =0
                                    ;16500          ;0------------------------------;
                                    ;16501          MDR_R[R1]_M[TEMP3],             ;
U 06DA, 0484,3592,4034,4467,0470,9  ;16502          PUSH,NEXT/FP.POLYD.60           ; RETURN FROM FP.POLYD.65
                                    ;16503
                                    ;16504          ;1------------------------------; RET FROM FP.MULDOP1 OF PART PRODUCT
                                    ;16505          M[TEMP2]_Q,                     ; BITS 0-31 TO WORKING TEMP
U 06DB, 0086,203A,403D,8047,0065,5  ;16506          NEXT/FP.POLYD.20                ; CONTINUE
```

```
                                    ;16507 =000
                                    ;16508 FP.POLYD.25:
                                    ;16509          ;000------------------------------; ARG NE 0
                                    ;16510          R[R2].SIZ_RB-1,CLEAR FLAG1,      ; R2 HAS DEGREE BYTE 0
                                    ;16511          SIZE[BYTE],SIGND CMP DEF?,       ; PCDELTA BYTE 2 FLAGS BYTE 3
U 0568, 080A,0E7D,0B44,8047,0865,5 407* ;16512          NEXT/FP.POLYD.20                 ; >=0,<0 OR TIMER
                                    ;16513
                                    ;16514          ;001------------------------------; ARG  EQ 0
                                    ;16515          WB_M[TEMP4],                     ; IS DEGREE REALLY 0
U 0569, 0880,4592,4A70,0047,0456,A  ;16516          WX.NE.0?,PUSH                    ; SPECIAL RETURN
                                    ;16517
                                    ;16518 FP.POLYD.30:
                                    ;16519          ;010------------------------------; DEGREE=0
                                    ;16520          R[R3]_M[VA],CLEAR FPD,           ; ARG=0,R4,R5=0 ALSO R0,R1 HAVE COEF
U 056A, 08E5,B592,4034,C047,006F,E  ;16521          NEXT/FP.POLYFD.END               ; R2 NEEDS 0 VA HAS VA+8
                                    ;16522
                                    ;16523          ;011------------------------------;
                                    ;16524          M[TEMP4]_MB-ZLIT0[1],            ; ADJUST FOR INCREMENTED VA
U 056B, 0D86,4C10,0030,0847,0470,0  ;16525          PUSH,NEXT/FP.POLYFD.BT30         ;
                                    ;16526
                                    ;16527          ;100------------------------------;
U 056C, 0884,4001,0031,0047,0056,D  ;16528          R[TEMP4]_M[TEMP4]+RB             ; IS DEGREE ALSO 0, *2 HERE AND *4 LATER
                                    ;16529
                                    ;16530 FP.POLYD.35:
                                    ;16531          ;101------------------------------; DEGREE NE 0 BUT ARG =0
                                    ;16532          VA_M[VA]+(R[TEMP4].ASL.SIZE),    ; TEMP4 HAS DEG*2 NOW GETS *8
                                    ;16533          SIZE[LONG],                      ;
U 056D, 0081,B5D1,0021,04A7,046F,B  ;16534          PUSH,NEXT/FP.POLYFD.RD           ; READ TO GET LAST COEFF
                                    ;16535
                                    ;16536 FP.POLYD.40:
                                    ;16537          ;110------------------------------;
                                    ;16538          R[R0]_M[TEMP1],                  ; SAVE 1ST PART OF COEF AND SET
U 056E, 0484,1592,4024,0847,0070,8  ;16539          CCOP1,SIZE[LONG]                 ; CONDITION CODES
                                    ;16540 =
                                    ;16541 FX.POLYGH.45:
                                    ;16542          ;-------------------------------;
                                    ;16543          R[R1]_M[MDR],                    ; SAVE PACKED BITS 32-63
U 0708, 0885,2592,40B4,4047,0000,1  ;16544          RETURN [+1]                      ;
                                    ;16545
                                    ;16546 FP.POLYD.60:
                                    ;16547          ;-------------------------------;
                                    ;16548          R[R2].SIZ_RB-1,                  ; DECREMENT DEGREE
                                    ;16549          SIZE[BYTE],                      ;
                                    ;16550          SIGND CMP DEF?,                  ;
U 0709, 0082,0E7D,0B44,8047,0866,1 407* ;16551          NEXT/FP.POLYD.90                 ; DONE YET
```

```
                                ;16552 =0
                                ;16553 FP.POLYD.65:
                                ;16554          ;0------------------------------;
                                ;16555          R[TEMP5]_EXP(M[TEMP1]),         ; UNPACK THE PART PRODUCT
                                ;16556          FRO.FLTZ?,                      ;
U 06DC, 0C84,1437,0AB1,4047,085E,4 323*         ;16557          NEXT/FP.MULDOP1                 ;
                                ;16558
                                ;16559          ;1------------------------------;
                                ;16560          D_ZLITO[0],                     ; CLEAR OUT D FOR FPD FAULT
                                ;16561          INTPEND OR TIMER?,              ;
                                ;16562          PUSH,                           ; CHECK IF INT OR TIMER
U 06DD, 0980,0C37,2AF0,0047,04F7,0      ;16563          NEXT/IE.SERV.IP.TS2             ; RETURN HERE ONLY IF NONE OR TIMER
                                ;16564 659:
                                ;16565 FREE.0659:
                                ;16566          ;01-----------------------------; VALUE IS 01XXXXXX...
                                ;16567          R[TEMP5].SIZ_RB-1,              ; DEC THE EXP
                                ;16568          SIZE[LONG],                     ;
                                ;16569          CLEAR READ(FLAG1),              ; CLEAR FLAG FOR READ
U 0659, 0C0A,0E7D,0021,4047,0070,C      ;16570          NEXT/FP.POLYD.80
                                ;16571 65B:
                                ;16572 FREE.065B:
                                ;16573          ;11-----------------------------;
U 065B, 0580,0EF6,4030,0847,0070,A      ;16574          PL_[1]                          ; SET UP PLATCH
                                ;16575 70A:
                                ;16576 FP.POLYD.72:
                                ;16577          ;-------------------------------;
U 070A, 0084,2137,0030,C047,0070,B      ;16578          R[TEMP3]_(M[TEMP2] RB).RR.P     ; SHIFT RIGHT 1
                                ;16579
                                ;16580          ;-------------------------------;
                                ;16581          M[TEMP2] (MB+R[ZERO]).SR.1,     ; BITS 0-31
                                ;16582          CLEAR READ(FLAG1),              ;
                                ;16583          MUL1(FLAG2) XOR MUL2(FLAG3)?,   ; WHAT IS PRODUCT SIGN
U 070B, 080E,2001,857D,8047,0065,E      ;16584          NEXT/FP.POLYD.75                ;
                                ;16585 13F8:
                                ;16586 FP.POLYD70.FIX:
                                ;16587          ;00-----------------------------; FRACTION IS 0
                                ;16588          M[TEMP5]_0,                     ; ZERO OUT EXP
U 13F8, 0886,55B7,0030,0047,0065,E      ;16589          NEXT/FP.POLYD.75                ; CONTINUE ON
                                ;16590
                                ;16591 13F9:
                                ;16592          ;01-----------------------------; VALUE IS 01XXXXXX...
                                ;16593          R[TEMP5].SIZ_RB-1,SIZE[LONG],   ; DEC THE EXP
                                ;16594          CLEAR READ(FLAG1),              ; CLEAR FLAG FOR READ
U 13F9, 0C0A,0E7D,0021,4047,0070,C      ;16595          NEXT/FP.POLYD.80
                                ;16596 13FA:
                                ;16597          ;10-----------------------------; VALUE IS 10XXXXXX..
                                ;16598          PL_[1],                         ; SET UP PLATCH
U 13FA, 0580,0EF6,4030,0847,0070,A      ;16599          NEXT/FP.POLYD.72                ; CONTINUE ON
                                ;16600
                                ;16601 13FB:
                                ;16602          ;11-----------------------------; VALUE IS 11XXXXXX..
                                ;16603          PL_[1],                         ; SET UP PLATCH
U 13FB, 0580,0EF6,4030,0847,0070,A      ;16604          NEXT/FP.POLYD.72                ; CONTINUE
```

```
                                ;16605 =10
                                ;16606 FP.POLYD.75:
                                ;16607          ;10-------------------------------; SIGN IS +
                                ;16608          CLEAR ADD1(FLAG0),              ; CLEAR THE SIGN FLAG
                                ;16609          READ,SIZE[LONG],                ;
                                ;16610          VA_VA+4,                        ;
U 065E, 0000,0036,4020,0450,0070,D ;16611       NEXT/FP.POLYD.100               ;
                                ;16612
                                ;16613 =11
                                ;16614          ;11-------------------------------; SIGN IS -
                                ;16615          SET ADD1(FLAG0),                ;
                                ;16616          READ,SIZE[LONG],                ;
                                ;16617          VA_VA+4,                        ;
U 065F, 0840,0036,4020,0450,0070,D ;16618       NEXT/FP.POLYD.100               ;
                                ;16619
                                ;16620 FP.POLYD.80:
                                ;16621          ;-------------------------------;
                                ;16622          M[TEMP3]_MB.AND.OLIT0[1FE],     ; STRIP OFF BIT 64
                                ;16623          MUL1(FLAG2) XOR MUL2(FLAG3)?,   ; WHAT IS PRODUCT SIGN
U 070C, 0D86,3E12,057F,F047,0065,E ;16624       NEXT/FP.POLYD.75                ;
                                ;16625 =01
                                ;16626 FP.POLYD.90:
                                ;16627          ;01-------------------------------; LOOP >=0 NOT DONE YET
                                ;16628          R[TEMP1]_M[TEMP2],              ;
                                ;16629          IP.TS?,                         ;
U 0661, 0884,2592,4B30,58E7,006D,C ;16630       NEXT/FP.POLYD.65                ;
                                ;16631
                                ;16632          ;11-------------------------------; DONE
U 0663, 0C84,05B7,0035,0047,0070,F ;16633       R[R4]_0,NEXT/FP.POLYD.220       ; ZERO OUT REGISTERS
                                ;16634
                                ;16635 FP.POLYD.100:
                                ;16636          ;-------------------------------;
                                ;16637          R[TEMP1].SIZ_Q_M[MDR],          ; BITS 0-31 INTO TEMP1 AND Q
U 070D, 084B,2592,5020,4047,0070,E ;16638       SIZE[LONG],SET READ(FLAG1)      ; SET FLAG FOR SECOND HALF READ
                                ;16639
                                ;16640          ;-------------------------------;
                                ;16641          READ,VA_VA+4,                   ; DO THE READ
                                ;16642          SIZE[LONG],CLEAR READ(FLAG1),   ;
U 070E, 0008,0036,4020,0450,013F,C ;16643       NEXT/FP.POLYD.FIX               ; RETURN IS TO FP.POLYD.22 +.011
                                ;16644
                                ;16645 FI.POLYD.220:
                                ;16646 FP.POLYD.220:
                                ;16647 FX.POLYD.220:
                                ;16648          ;-------------------------------;
                                ;16649          R[R5]_0,                        ; ZERO OUT SECOND HALF OF ARG
U 070F, 0084,05B7,0035,4047,0056,A ;16650       NEXT/FP.POLYD.30                ; CLEAN UP AND FINISH UP
```

```
                                ;16651 13FC:
                                ;16652 FP.POLYD.FIX:
                                ;16653          ;0------------------------------;
                                ;16654          R[TEMP7] Q_EXP(M[TEMP1]),       ; SAVE EXPONENT IN Q ALSO
                                ;16655          FRO.FLTZ?,SET FLAG3,            ; TEST VALIDITY OF OPERAND
U 13FC, 085C,1435,1AB1,C047,0C5C,8 389*  ;16656          PUSH,NEXT/FP.ADDDOP2            ; FLAG3 IS CLEARED IF OP=0
                                ;16657
                                ;16658 13FD:
                                ;16659          ;1------------------------------; MUST TEST FOR ZERO
                                ;16660          OPZERO(FLAG3)?,                 ; 0 COEFFICIENT?
U 13FD, 0080,0036,44F0,0047,013F,E       ;16661          NEXT/FP.POLYD.FIX2              ;
                                ;16662
                                ;16663 13FE:
                                ;16664 FP.POLYD.FIX2:
                                ;16665          ;0------------------------------; EQ 0 SKIP ADDDRTN ROUTINE
                                ;16666          M[TEMP5] MB-ZLITO[1E],          ; SUB THE FUDGE FACTOR
                                ;16667          ADD1(FLAG0)?,                   ; SIGN OF PART PRODUCT
U 13FE, 0586,5C10,0430,F047,0050,A       ;16668          NEXT/FP.NORM.MUD1               ; OFF TO SET D CORRECTLY
                                ;16669                                          ; RET TO FP.POLYD.22+.011
                                ;16670
                                ;16671 13FF:
                                ;16672          ;1------------------------------; NE 0 TO ADDDRTN ROUTINE
U 13FF, 0480,0036,4030,0047,000A,3       ;16673          NEXT/FP.ADDD.25                 ; RET TO FP.POLYD.22+.011
```

```
;16674 .TOC "  Floating point and CRC          : FPA INTERFACE    "
;16675
;16676 ;*****************************************************************************
;16677 ;
;16678 ;       FPA INTERFACE ROUTINES
;16679 ;
;16680 ;               ENTRY                           INSTRUCTION
;16681 ;               -----                           -----------
;16682 ;
;16683 ;               FI.ADDF2.REG/FI.ADDF2.MEM       ADDF2,SUBF2,MULF2,DIVF2
;16684 ;
;16685 ;               FI.ADDF3.REG/FI.ADDF3.MEM       ADDF3,SUBF3,MULF3,DIVF3
;16686 ;                                               CVTBF,CVTWF,CVTLF
;16687 ;                                               CVTFB,CVTFW,CVTFL,CVTRFL
;16688 ;                                               CVTDB,CVTDW,CVTDL,CVTRDL
;16689 ;
;16690 ;               FI.ADDD2.REG/FI.ADDD2.MEM       ADDD2,SUBD2,MULD2,DIVD2
;16691 ;
;16692 ;               FI.ADDD3.REG/FI.ADDD3.MEM       ADDD3,SUBD3,MULD3,DIVD3
;16693 ;                                               CVTBD,CVTWD,CVTLD
;16694 ;
;16695 ;               FI.CMPF.REG/FI.CMPF.MEM         CMPF
;16696 ;
;16697 ;               FI.CMPD.REG/FI.CMPF.MEM         CMPD
;16698 ;
;16699 ;               FI.CVTDF                        CVTDF
;16700 ;
;16701 ;               FI.CVTFD                        CVTFD
;16702 ;
;16703 ;               FI.EMODF                        EMODF
;16704 ;
;16705 ;               FI.EMODD                        EMODD
;16706 ;
;16707 ;               FI.POLYF                        POLYF
;16708 ;
;16709 ;               FI.POLYF.FPD.RES                POLYF RESTART
;16710 ;
;16711 ;               FI.POLYD                        POLYD
;16712 ;
;16713 ;               FI.POLYD.FPD.RES                POLYD RESTART
;16714 ;
;16715 ;*****************************************************************************
```

```
                        ;16716 .TOC "  Floating point and CRC          : FPA INTERFACE   "
                        ;16717 .TOC "  Floating point and CRC          :       FI.ADDF2.REG/.MEM,FI.ADDF3.REG/.MEM
                        ;16718 .REGION/IRD1.R1L,IRD1.R1H
                        ;16719
                        ;16720 FI.ADDF2.REG:                          ; **********************
                        ;16721          ;-------------------------------;
                        ;16722          FPA_M[MDR] FPA_WB_R[GPR.R]-0,  ; 1ST OP (MBUS) 2ND OP (WB) REG
U 035C, 0481,25BC,003C,C047,300D,D      ;16723          NEXT/FI.ADDF2.20               ; WAIT AND WRITE
                        ;16724
                        ;16725 FI.ADDF2.MEM:                          ; **********************
                        ;16726          ;-------------------------------;
                        ;16727          FPA_M[MDR],                    ; 2ND OP (MBUS) MEM
U 035D, 0C81,2036,4030,0047,100D,D      ;16728          NEXT/FI.ADDF2.20               ; WAIT AND WRITE
                        ;16729
                        ;16730 =0
                        ;16731 FI.ADDF3.REG:                          ; **********************
                        ;16732          ;0------------------------------;
                        ;16733          FPA_M[MDR] FPA_WB_R[GPR.R]-0,  ; 1ST OP (MBUS) 2ND OP (WB) REG
                        ;16734          PUSH,LOD INC BRA?,             ; GET DESTINATION
U 00DC, 0C81,25BC,01BC,C047,3414,0      ;16735          NEXT/OS.WRT1                   ;
                        ;16736
                        ;16737 FI.ADDF2.20:                           ; ENTRY FROM IRDROM
                        ;16738          ;1-------------------------------;
                        ;16739          FPAWAIT,                       ; STALL FOR FPA DONE
                        ;16740          R[DST.R].SIZ_FPA,              ; WRITE FROM FPA
                        ;16741          WRITE NOTREG,SIZE[IDEP],       ;
U 00DD, 0E82,0036,413C,4DDA,403F,9      ;16742          CCOP1,IRD1                     ; SET CONDITION CODES EXIT
                        ;16743 =0
                        ;16744 FI.ADDF3.MEM:                          ; **********************
                        ;16745          ;0-------------------------------;
                        ;16746          FPA_M[MDR],                    ; 2ND OP (MBUS) MEM
                        ;16747          PUSH,LOD INC BRA?,             ; GET DESTINATION
U 00E6, 0481,2036,41B0,0047,1414,0      ;16748          NEXT/OS.WRT1                   ;
                        ;16749
                        ;16750 FI.ADDF3.20:                           ; ENTRY FROM IRDROM
                        ;16751          ;1-------------------------------;
                        ;16752          FPAWAIT,                       ; STALL FOR FPA DONE
                        ;16753          R[DST.R].SIZ_FPA,              ; WRITE FROM FPA
                        ;16754          WRITE NOTREG,SIZE[IDEP],       ;
U 00E7, 0E82,0036,413C,4DDA,403F,9      ;16755          CCOP1,IRD1                     ; SET CONDITION CODES EXIT
```

```
                                ;16756 .TOC "  Floating point and CRC          :       FI.ADDD2.REG/.MEM    "
                                ;16757 .REGION/IRD1.R1L,IRD1.R1H
                                ;16758 =0
                                ;16759 FI.ADDD2.REG:                            ; **********************
                                ;16760          ;0------------------------------;
                                ;16761          FPA_M[MDR] FPA_WB_R[GPR.R]-0,   ; 2ND HALF 1ST OP (MBUS)
U 00EC, 0C81,25BC,003C,C047,3435,F ;16762       PUSH,NEXT/FI.ADDD2.100          ; 1ST HALF 2ND OP (WB) REG
                                ;16763
                                ;16764 FI.ADDD2.20:                             ; ENTRY FROM IRDROM
                                ;16765          ;1------------------------------;
                                ;16766          FPAWAIT,                        ; STALL FOR FPA
                                ;16767          R[DST.R].SIZ_FPA,               ; WRITE OUT 1ST HALF OF ANSWER
                                ;16768          SET MM.NOINT,                   ; NO INTERRUPTS ALLOWED
                                ;16769          WRITE NOTREG,SIZE[IDEP],        ;
                                ;16770          CCOP1,                          ; SET CONDITION CODES
U 00ED, 0A62,0036,493C,4DDA,400F,2 ;16771       REG MODE?,NEXT/FI.ADDD2.40      ; REGISTER OR MEMORY DESTINATION
                                ;16772
                                ;16773 =0
                                ;16774 FI.ADDD2.40:
                                ;16775          ;0------------------------------; MEMORY SO BUMP UP VA
U 00F2, 0C80,0036,4030,0447,000F,3 ;16776       VA_VA+4                         ; BUMP UP VA
                                ;16777
                                ;16778          ;1------------------------------;
                                ;16779          FPAWAIT,                        ; STALL AGAIN FOR FPA
                                ;16780          R[DST.R+1]_FPA,                 ; WRITE SECOND HALF OF ANSWER
                                ;16781          WRITE NOTREG,                   ;
U 00F3, 0E84,0036,412F,45DA,403F,9 ;16782       SIZE[LONG],IRD1                 ; END OF INSTRUCTION
                                ;16783
                                ;16784 FI.ADDD2.MEM:                            ; **********************
                                ;16785          ;------------------------------;
                                ;16786          FPA_M[MDR],                     ; 2ND HALF 2ND OP
U 035E, 0C81,2036,4030,0047,100E,D ;16787       NEXT/FI.ADDD2.20                ; STALL AND WRITE FROM FPA
                                ;16788
                                ;16789 FI.ADDD2.100:
                                ;16790          ;------------------------------;
                                ;16791          FPA_WB_R[GPR.R+1]-0,            ; 2ND HALF 2ND OP (WB) REG
U 035F, 0080,05BC,00BF,C047,6800,1 397* ;16792  RETURN[+1]                       ;
```

```
                        ;16793 .TOC "  Floating point and CRC          :       FI.ADDD3.REG/.MEM    "
                        ;16794
                        ;16795 FI.ADDD3.REG:                          ; ********************
                        ;16796          ;0-----------------------------;
 0367, 0481,258C,003C,C047,300F,C       ;16797          FPA_M[MDR] FPA_WB_R[GPR.R]-0  ; 2ND HALF 1ST OP (MBUS)
                        ;16798                                        ; 1ST HALF 2ND OP (WB) REG
                        ;16799
                        ;16800 =0
                        ;16301          ;0-----------------------------;
                        ;16802          FPA_WB_R[GPR.R+1]-0,          ; 2ND HALF 2ND OP REG
                        ;16803          PUSH,LOD INC BRA?,            ; GET DESTINATION
 00FC, 0480,056C,015F,C047,6C14,0 397*  ;16804          NEXT/OS.WRT1                  ;
                        ;16805
                        ;16806          ;1-----------------------------;
                        ;16807          FPAWAIT,                      ; STALL FOR FPA
                        ;16808          R[DST.R].SIZ_FPA,             ; WRITE OUT 1ST HALF OF ANSWER
                        ;16809          SET MM.NOINT,                 ; NO INTERRUPTS ALLOWED
                        ;16810          WRITE NOTREG,SIZE[IDEP],      ;
                        ;16811          CCOP1,                        ; SET CONDITION CODES
 00FD, 0A62,0036,493C,4DDA,400F,2       ;16812          REG MODE?,NEXT/FI.ADDD2.40    ; REGISTER OR MEMORY DESTINATION
                        ;16813 =0
                        ;16814 FI.ADDD3.MEM:                          ; ********************
                        ;16815          ;0-----------------------------;
                        ;16816          FPA_M[MDR],                   ; 2ND HALF 2ND OP
                        ;16817          PUSH,LOD INC BRA?,            ; GET DESTINATION
 01DA, 0481,2056,41B0,C047,1414,0       ;16818          NEXT/OS.WRT1                  ;
                        ;16819
                        ;16820 FI.ADDD3.20:
                        ;16821          ;1-----------------------------;
                        ;16822          FPAWAIT,                      ; STALL FOR FPA
                        ;16823          R[DST.R].SIZ_FPA,             ; WRITE OUT 1ST HALF OF ANSWER
                        ;16824          SET MM.NOINT,                 ; NO INTERRUPTS ALLOWED
                        ;16825          WRITE NOTREG,SIZE[IDEP],      ;
                        ;16826          CCOP1,                        ; SET CONDITION CODES
 01DB, 0A62,0036,493C,4DDA,400F,2       ;16827          REG MODE?,NEXT/FI.ADDD2.40    ; REGISTER OR MEMORY DESTINATION
```

```
                        ;16828 .TOC "  Floating point and CRC          :       FI.CMPF.REG/.MEM,FI.CMPD.REG    "
                        ;16829 .REGION/IRD1.R1L,IRD1.R1H
                        ;16830
                        ;16831 FI.CMPF.REG:                            ; ********************
                        ;16832         ;--------------------------------;
                        ;16833         FPA_M[MDR] FPA_WB_R[GPR.R]-0,   ; 1ST OP (MBUS) 2ND OP (WB) REG
U 036B, 0C81,25BC,003C,C047,301E,7      ;16834         NEXT/FI.CMPD.100                ; CONTINUE ON
                        ;16835
                        ;16836 FI.CMPF.MEM:                            ; ********************
                        ;16837         ;--------------------------------;
                        ;16838         FPA_M[MDR],                     ; 2ND OP MEM
U 036C, 0481,2036,4030,0047,101E,7      ;16839         NEXT/FI.CMPD.100                ; CONTINUE ON
                        ;16840
                        ;16841 =0
                        ;16842 FI.CMPD.REG:                            ; ********************
                        ;16843         ;0-------------------------------;
                        ;16844         FPA_M[MDR] FPA_WB_R[GPR.R]-0,   ; 2ND HALF 1ST OP (MBUS)
U 01E6, 0C81,25BC,003C,C047,3435,F      ;16845         PUSH,NEXT/FI.ADD2.100           ; 1ST HALF 2ND OP (WB) REG
                        ;16846
                        ;16847 FI.CMPD.100:
                        ;16848         ;1-------------------------------;
                        ;16849         WX_CONX.SIZ,                    ; FORCE WX.NE.0 FOR CCOP2 (NOT WBUS)
                        ;16850         FPAWAIT.SIZE[IDEP],             ; STALL FOR FPA
                        ;16851         WB_FPA,CCOP2,                   ; SET CCS
U 01E7, 0280,0736,4130,1047,403F,9      ;16852         IRD1                            ; END OF INSTRUCTION
```

```
                        ;16853 .TOC "  Floating point and CRC          :       FI.CVTDF,FI.CVTFD    "
                        ;16854 .REGION/IRD1.R1L,IRD1.R1H
                        ;16855
                        ;16856 FI.CVTDFD:                              ; **********************
                        ;16857          ;-------------------------------; 1ST OP (CVTFD)
                        ;16858          FPA_M[MDR],                    ; 2ND HALF OP (CVTDF)
                        ;16859          LOD INC BRA?,                  ; GET DESTINATION
U 036D, 0C81,2036,41B0,0047,1014,0      ;16860          NEXT/OS.WRT1                  ;
                        ;16861
                        ;16862 FI.CVTDF.10:                            ; ENTRY THROUGH IRDROM (CVTDF)
                        ;16863          ;-------------------------------;
                        ;16864          FPAWAIT,                       ; STALL FOR FPA
                        ;16865          R[DST.R] FPA,                  ; WRITE OUT ANSWER
                        ;16866          WRITE NOTREG,SIZE[IDEP],       ;
                        ;16867          SET MM.NOINT,                  ; NO INTERRUPTS
U 036E, 0E64,0036,403C,45DA,401E,D      ;16868          NEXT/FI.CVTDFD.50              ; GET THE CCS FROM FPA
                        ;16869
                        ;16870 =0
                        ;16871 FI.CVTFD.10:                            ; ENTRY THROUGH IRDROM (CVTFD)
                        ;16872          ;0-------------------------------;
                        ;16873          FPAWAIT,                       ; STALL FOR FPA
                        ;16874          R[DST.R] FPA,                  ; WRITE OUT 1ST HALF OF ANSWER
                        ;16875          WRITE NOTREG,SIZE[IDEP],       ;
                        ;16876          SET MM.NOINT,REG MODE?,        ; NO INTERRUPTS ALLOWED
U 01EC, 0E64,0036,493C,45DA,441F,2      ;16877          PUSH,NEXT/FI.CVTFD.40          ; WRITE SECOND HALF
                        ;16878
                        ;16879 FI.CVTDFD.50:
                        ;16880          ;1-------------------------------;
                        ;16881          FPAWAIT,                       ; WAIT FOR CCS FROM FPA
                        ;16882          CC_FPA,                        ; WRITE THEM OUT
U 01ED, 0280,0036,4130,00A7,503F,9      ;16883          IRD1                          ; END OF INSTRUCTION
                        ;16884
                        ;16885 =0
                        ;16886 FI.CVTFD.40:
                        ;16887          ;0-------------------------------; NOT REG MODE
U 01F2, 0480,0036,4030,0447,001F,3      ;16888          VA_VA+4                        ; BUMP VA
                        ;16889
                        ;16890          ;1-------------------------------;
                        ;16891          FPAWAIT,                       ; STALL FOR FPA AGAIN
                        ;16892          R[DST.R+1] FPA,                ; WRITE BITS 32-63
                        ;16893          WRITE NOTREG,SIZE[LONG],       ;
U 01F3, 0684,0036,40AF,45DA,4000,1      ;16894          RETURN [+1]                    ;
```

```
                        ;16895 .TOC "  Floating point and CRC          :       FI.EMODF    "
                        ;16896 .REGION/IRD1.R1L,IRD1.R1H
                        ;16897
                        ;16898 =00
                        ;16899 FI.EMODF:                                ; **********
                        ;16900          ;00-----------------------------;
                        ;16901          LOD INC BRA?,                   ; GET THIRD OP
U 00F4, 0480,0036,41B0,0047,0410,0      ;16902          PUSH,NEXT/OS.RED                ;
                        ;16903
                        ;16904          ;01-----------------------------;
                        ;16905          FPA_M[MDR],                     ; WRITE OP3 MULD.RF
                        ;16906          LOD INC BRA?,                   ; GET INT.WL
U 00F5, 0481,2036,41B0,0047,1414,0      ;16907          PUSH,NEXT/OS.WRT1               ;
                        ;16908
                        ;16909          ;10-----------------------------;
                        ;16910          R[TEMP10]_M[VA],                ; SAVE ADDRESS OF INT.WL
                        ;16911          REG MODE?,                      ; WAS IT IN REGISTER
U 00F6, 0485,B592,4932,8047,0453,E      ;16912          PUSH,NEXT/FI.EMODF.300          ; CHECK IN MAIN FLOWS
                        ;16913
                        ;16914          ;11-----------------------------;
                        ;16915          R[TEMP1]_RNUM,                  ; NOW HAVE FRACT.WF ALSO
                        ;16916          REG MODE?,                      ; WAS IT REGISTER
U 00F7, 0085,6036,4930,4047,0020,2      ;16917          NEXT/FI.EMODF.100               ; IF NOT MUST PROBE
                        ;16918
                        ;16919 =0
                        ;16920 FI.EMODF.100:
                        ;16921          ;0------------------------------; MEMORY MUST PROBE
                        ;16922          R[TEMP1]_M[VA],                 ; SAVE VA INS EAD
U 0202, 0885,B592,4030,4047,0579,6      ;16923          PUSH,NEXT/MM.PRB.WRITE.SIZ.00   ; RET +1 IF O. 4Y
                        ;16924
                        ;16925          ;1------------------------------; REG MODE OR OKAY
                        ;16926          VA_RNUM_R[TEMP10],              ; RESTORE POINTERS FOR INT.WL
U 0203, 0C81,D5BE,45F2,84A7,0020,C      ;16927          REGINT(FLAG1)?                  ; FLAG FOR REGMODE FOR INT.WL
                        ;16928 =0*
                        ;16929          ;0*-----------------------------;
                        ;16930          FPAWAIT,                        ; STALL FOR FPA
                        ;16931          WRITE, WB_FPA,                  ; WRITE TO MEMORY
                        ;16932          SET MM.NOINT,SIZE[LONG],        ; NO INTERRUPTS
U 020C, 0A60,0036,4020,05D8,4036,F      ;16933          NEXT/FI.EMODF.150               ; CONTINUE WITH FRACT.WF
                        ;16934
                        ;16935          ;1*-----------------------------; WRITE TO REGISTER
                        ;16936          FPAWAIT,                        ; STALL FOR FPA
                        ;16937          R[GPR.R]_FPA,                   ; TO REGISTER
                        ;16938          SET MM.NOINT,SIZE[LONG],        ;
U 020E, 0E64,0036,402C,C047,4036,F      ;16939          NEXT/FI.EMODF.150               ;
```

```
                                ;16940 FI.EMODF.150:
                                ;16941          ;--------------------------------;
                                ;16942          VA_RNUM_R[TEMP1],               ; GET READY TO WRITE FRACT.WX
U 036F, 0C81,D5BE,48F0,44A7,0021,2      ;16943          IR<5>?                          ; WAS IT EMODF OR EMODD
                                ;16944
                                ;16945 =0
                                ;16946          ;0--------------------------------; EMODF
                                ;16947          FPAWAIT,                        ; WAIT FOR FPA
                                ;16948          R[DST.R]_FPA,                   ; WRITE OUT FRACT.WF
                                ;16949          WRITE NOTREG,SIZE[IDEP],        ;
U 0212, 0E84,0036,413C,4DDA,403F,9      ;16950          CCOP1,IRD1                      ; DONE AT LAST
                                ;16951
                                ;16952          ;1--------------------------------; EMODD
                                ;16953          FPAWAIT,                        ; STALL FOR FPA
                                ;16954          R[DST.R]_FPA,                   ; WRITE OUT 1ST HALF
                                ;16955          WRITE NOTREG,SIZE[IDEP],        ;
                                ;16956          CCOP1,REG MODE?,                ; GO WRITE 2ND HALF
U 0213, 0284,0036,493C,4DDA,400F,2      ;16957          NEXT/FI.ADDD2.40                ; AND END
```

```
                                ;16958 .TOC "  Floating point and CRC          :       FI.EMODD    "
                                ;16959 .REGION/IRD1.R1L,IRD1.R1H
                                ;16960
                                ;16961 =00
                                ;16962 FI.EMODD:
                                ;16963          ;00-------------------------------;
                                ;16964          LOD INC BRA?,                     ; GET MULD.RD
U 00F8, 0C80,0036,41B0,0C47,041B,0  ;16965      PUSH,NEXT/OS.FIDRED               ;
                                ;16966
                                ;16967          ;01-------------------------------;
                                ;16968          FPA M[MDR],                       ; WRITE BITS 32-63 OF MULD.RD
                                ;16969          PUSH,LOD INC BRA?,                ; GET INT.WL
U 00F9, 0481,2036,41B0,0047,1414,0  ;16970      NEXT/OS.WRT1                      ;
                                ;16971
                                ;16972          ;10-------------------------------;
                                ;16973          R[TEMP10]_M[VA],                  ; SAVE VA
                                ;16974          REG MODE?,                        ;
U 00FA, 0485,B592,4932,8047,0453,E  ;16975      PUSH,NEXT/FI.EMODF.300            ; CHECK IF REG MODE GET FRACT.WD
                                ;16976
                                ;16977          ;11-------------------------------;
                                ;16978          R[TEMP1]_RNUM,                    ; SAVE FRACT.WD IF RNUM
                                ;16979          REG MODE?,                        ; IS IT
U 00FB, 0085,6036,4930,4047,0020,2  ;16980      NEXT/FI.EMODF.100                 ; JOIN COMMON FLOWS TO END
```

```
                                    ;16981 .TOC "  Floating point and CRC          :       FI.POLYF    "
                                    ;16982 .REGION/IRD1.R1L,IRD1.R1H
                                    ;16983 =0
                                    ;16984 FI.POLYF:                                ; **********************
                                    ;16985          ;0------------------------------;
                                    ;16986          M[TEMP2] Q,                     ; SAVE ARG IN TEMP TILL 1ST PP READY
U 0222, 0886,203A,403D,8047,0422,C  ;16987          PUSH,NEXT/FI.POLYFD.10          ; DEG TO TEMP GET ADD
                                    ;16988
                                    ;16989          ;1------------------------------; DEGREE > 31 TEST
                                    ;16990          WB_M[TEMP4].ANDNOT.ZLIT0[1F],   ;
U 0223, 0180,4C13,8A70,F847,001D,C  ;16991          WX.NE.0?                        ;
                                    ;16992
                                    ;16993 =00      ;00------------------------------; DEGREE OK
                                    ;16994          R[TEMP4].SIZ_RB-1,              ; DECREMENT DEGREE
                                    ;16995          SIZE[BYTE],MDR_0,               ; ZERO MDR FOR DEG =0
                                    ;16996          SIGND CMP?,                     ;
U 01DC, 0C82,0E7D,0B41,04E7,0C23,8 407* ;16997      PUSH,NEXT/FI.POLYF.30           ; RET IS +3
                                    ;16998
                                    ;16999          ;01-----------------------------; DEGREE >31 FAULT
U 01DD, 0C80,0036,4030,0047,00FF,8  ;17000          NEXT/IE.OPER.FAULT              ;
                                    ;17001
                                    ;17002 =11
                                    ;17003          ;11-----------------------------;
                                    ;17004          R[R1] 0,                        ; START CLEAN UP
U 01DF, 0484,05B7,0034,4047,0063,0  ;17005          NEXT/FI.POLYF.20                ; GO TO END UP
                                    ;17006 =000
                                    ;17007 FI.POLYF.30:
                                    ;17008          ;000-----------------------------; DEGREE >1
                                    ;17009          READ,VA_VA+4,SIZE[LONG],        ; READ 1ST COEF
U 0238, 0880,0036,4020,0450,0423,A  ;17010          PUSH,NEXT/FI.POLYFD.RW3.5       ; DO TRANS TO FPA  GET ANS (RET+3)
                                    ;17011
                                    ;17012          ;001----------------------------; DEGREE = 1
                                    ;17013          READ,VA_VA+4,                   ; READ COEF AND FALL THROUGH TO FINISH
U 0239, 0080,0036,4020,0450,0023,A  ;17014          SIZE[LONG]                      ;
                                    ;17015
                                    ;17016 FI.POLYFD.RW3.5:
                                    ;17017          ;010----------------------------; DEGREE =0 ANS IS LAST COEF
                                    ;17018          FPA_M[MDR],READ,VA_VA+4,        ; SEND 0 TO NEGATE ARG IF DEG =0
                                    ;17019          SIZE[LONG],CLEAR READ(FLAG1),   ;
U 023A, 0809,2036,4020,0450,1037,A  ;17020          NEXT/FI.POLYFD.RW4              ; TRANS TO FPA
                                    ;17021                                          ; GET ANS RET +3 TO PREVIOUS PUSH
                                    ;17022
                                    ;17023
                                    ;17024          ;011----------------------------; RET +3 FROM DEG>1
                                    ;17025          R[R1] M[TEMP2],                 ; SAVE ARG IN REG
U 023B, 0084,2592,4034,4047,047F,C  ;17026          PUSH,NEXT/FI.POLYFD.SET         ; DO FPD SET UP
                                    ;17027
                                    ;17028 FI.POLYF.50:
                                    ;17029          ;100----------------------------;
                                    ;17030          R[R2].SIZ_RB-1,                 ; DECREMENT DEG COUNT
                                    ;17031          SIZE[BYTE],                     ;
                                    ;17032          CLEAR FPA(FLAG0),               ; CLEAR FPA FAULT FLAG
                                    ;17033          COUNT OR INT TIMER?,            ;
U 023C, 0C02,0E7D,0B04,8047,081E,8 410* ;17034      NEXT/FI.POLYF.70                ;
                                    ;17035 =
```

```
                                        ;17036 =00
                                        ;17037 FI.POLYF.70:
                                        ;17038          ;00-------------------------------; LOOP >=0 NO INT
                                        ;17039          READ,VA VA+4,                     ; GET NEXT COEF
                                        ;17040          SIZE[LONG],                       ;
U 01E8, 0880,0036,4020,0450,0437,A      ;17041          PUSH,NEXT,FI.POLYFD.RW4           ;
                                        ;17042
                                        ;17043          ;01-------------------------------; LOOP >=0 INT OR TIMER
                                        ;17044          INTPEND OR TIMER?,PUSH,           ; CHECK WHICH
U 01E9, 0480,0036,4AF0,0047,04F7,0      ;17045          NEXT/IE.SERV.IP.TS2               ; RET -1 IF TIMER ONLY
                                        ;17046
                                        ;17047          ;10-------------------------------; LOOP <0 DONE
                                        ;17048          R[R1] 0,                          ; FINISH UP
U 01EA, 0484,05B7,0034,4047,0063,0      ;17049          NEXT/FI.POLYF.20                  ; MAIN FLOWS
                                        ;17050
                                        ;17051          ;11-------------------------------; RET +3 FROM FPA TRANS GET ANS
                                        ;17052          R[R2].SIZ_RB-1,                   ; DECREMENT DEG COUNT
                                        ;17053          SIZE[BYTE],                       ;
                                        ;17054          CLEAR FPA(FLAG0),                 ; CLEAR FPA FAULT FLAG
                                        ;17055          COUNT OR INT TIMER?,              ;
U 01EB, 0C02,0E7D,0B04,8047,081E,8 410* ;17056          NEXT/FI.POLYF.70                  ;
```

```
                                ;17057 .TOC "  Floating point and CRC          :       FI.POLYF & FI.POLYD SUBROUTINES     "
                                ;17058 ;*****************************************************************************
                                ;17059 ;
                                ;17060 ;       FI.POLYF AND FI.POLYD SUBROUTINES
                                ;17061 ;
                                ;17062 ;
                                ;17063 ;
                                ;17064 ;*****************************************************************************
                                ;17065
                                ;17066 .REGION/IRD1.R1L,IRD1.R1H
                                ;17067 =0
                                ;17068 FI.POLYFD.10:                            ; ENTRY FROM IRDROM POLYD FPA
                                ;17069         ;0-------------------------------;
                                ;17070         R[TEMP4]_ZEXT(M[MDR]),          ; ZERO EXTEND DEGREE
                                ;17071         SIZE[WORD],LOD INC BRA?,        ; GET TBL ADDR
U 022C, 0C85,259E,4191,0047,0412,0 ;17072      PUSH,NEXT/OS.ADD                ;
                                ;17073
                                ;17074         ;1-------------------------------;
                                ;17075         VA_M[MDR],                      ; PUT ADDRESS IN VA FREE UP MDR
U 022D, 0C81,2002,40BD,84A7,0000,1 ;17076      RETURN [+1]                     ; RETURN TO CALLING INSTRUCTION
                                ;17077 07E6:
                                ;17078 FI.POLYFD.RW1:
                                ;17079         ;*********FORCE ADDRESS*********; FLAG0 AND FLAG1 ARE CLEAR UPON ENTRY
                                ;17080         FPA_M[MDR],CLEAR READ(FLAG1),   ; WRITE 0-31 FIRST COEF
U 07E6, 0009,2036,4020,0450,1037,7 ;17081      READ,VA_VA+4,SIZE[LONG]         ; READ 32-63 FIRST COEF
                                ;17082
                                ;17083 FI.POLYFD.RW2:
                                ;17084         ;-------------------------------;
U 0377, 0481,2036,4030,0047,1037,9 ;17085      FPA_M[MDR]                      ; SEND TO FPA 32-63
                                ;17086
                                ;17087 FI.POLYFD.RW3:
                                ;17088         ;-------------------------------;
                                ;17089         READ,VA_VA+4,SIZE[LONG],        ; GET 2ND COEF
                                ;17090         FPAWAIT,SET READ(FLAG1),        ; SET 2ND HALF READ FLAG
U 0379, 0248,0036,4020,0450,0023,A ;17091      NEXT/FI.POLYFD.RW3.5            ;
                                ;17092
                                ;17093 FI.POLYFD.RW4:
                                ;17094         ;-------------------------------;
                                ;17095         FPA_M[MDR],                     ; SEND 2ND HALF COEF OR COEF IF POLYF
U 037A, 0641,2036,4030,0047,107E,C ;17096      FPAWAIT,SET FPA(FLAG0)          ; SIGNAL ANY FAULTS FOUND BY FPA
                                ;17097
                                ;17098 07EC:    ;*********FORCE ADDRESS*********;
                                ;17099         FPAWAIT,                        ; STALL FOR FPA
                                ;17100         R[R0]_FPA,SIZE[LONG],           ; WRITE OUT 1ST HALF PART PRODUCT
                                ;17101         CCOP1,CLEAR FPA(FLAG0),         ; OR PART PRODUCT IF POLYF
U 07EC, 0A04,0036,40A4,0847,4000,3 ;17102      RETURN [+3]                     ; RETURN TO CALLING ROUTINE
```

```
                                ;17103 07FC:
                                ;17104 FI.POLYFD.SET:
                                ;17105          ;0********FORCE ADDRESS*********;
                                ;17106          PUSH,IR<5>?,                    ; GET FPDOFFSET SET
U 07FC, 0880,0036,48F0,0047,0423,E ;17107       NEXT/FI.POLYFD.OFF              ;
                                ;17108
                                ;17109 07FD:    ;1********FORCE ADDRESS*********;
                                ;17110          R[R3]_M[VA],                    ; SAVE VA
U 07FD, 0885,B592,4034,C047,006D,5 ;17111       NEXT/FI.PCDELTA                 ; R2 AND FPD SET RETURN TO CALLING RTN
                                ;17112
                                ;17113 =0
                                ;17114 FI.POLYF.OFF:
                                ;17115 FI.POLYFD.OFF:
                                ;17116          ;0-----------------------------; FOR POLYF
                                ;17117          M[FPDOFFSET]_ZLITO[12.],        ;
U 023E, 0D86,CC37,00B0,6047,0000,1 ;17118       RETURN [+1]                     ;
                                ;17119
                                ;17120 FI.POLYD.OFF:
                                ;17121          ;1-----------------------------; FOR POLYD
                                ;17122          M[FPDOFFSET]_ZLITO[13.],        ;
U 023F, 0986,CC37,00B0,6847,0000,1 ;17123       RETURN [+1]                     ;
```

```
                                ;17124 .TOC "  Floating point and CRC          :       FI.POLYF & FI.POLYD FPD SAVE ROUTINE
                                ;17125 ;*****************************************************************************
                                ;17126 ;
                                ;17127 ;       POLYF AND POLYD FIRST PART DONE ROUTINES FOR FPA INTERFACE
                                ;17128 ;
                                ;17129 ;               Pack up routines
                                ;17130 ;                       Initial instruction at FPDOFFSET #12 for POLYF
                                ;17131 ;                                             at FPDOFFSET #13 for POLYD
                                ;17132 ;                       THIS IS CODE THAT APPEARS IN IANDE
                                ;17133 ;                                       IN FPDOFFSET #12 FOR  POLYF
                                ;17134 ;                                       IN FPDOFFSET #13 FOR  POLYD
                                ;17135 ;
                                ;17136 ;FI.POLYF.FPD.SAVE:
                                ;17137 ;       ;--------------------------------;
                                ;17138 ;       R[R2].SIZ_RB+1,                 ; INCREMENT ITERATION COUNT
                                ;17139 ;       SIZE[BYTE],                     ;
                                ;17140 ;       FPA(FLAG0)?,                    ; FAULT DETECTED BY FPA?
                                ;17141 ;       NEXT/FI.POLYF.FPD.10            ; CONTINUE ON
                                ;17142 ;
                                ;17143 ;FI.POLYD.FPD.SAVE:
                                ;17144 ;       ;--------------------------------;
                                ;17145 ;       R[R2].SIZ_RB+1,                 ; INCREMENT ITERATION COUNT
                                ;17146 ;       SIZE[BYTE],                     ; SECOND HALF READ OR
                                ;17147 ;       READ(FLAG1) FPA(FLAG0)?,        ; FAULT DETECTED BY FPA?
                                ;17148 ;       NEXT/FI.POLYD.FPD.10            ; CONTINUE ON
                                ;17149 ;
                                ;17150 ;*****************************************************************************
                                ;17151
                                ;17152 =0
                                ;17153 FI.POLYF.FPD.10:
                                ;17154         ;0------------------------------;
                                ;17155         R[R3]_M[VA],                    ; SAVE VA
U 024C, 0085,B592,4034,C047,00FE,2 ;17156      NEXT/IE.PACK.DONE               ; THATS ALL FOR NOW
                                ;17157
                                ;17158         ;1------------------------------;
                                ;17159         VA_M[VA]-ZLITO[4],              ; DECREMENT VA
U 024D. 0581,BC10,0030,24A7,0024,C ;17160      NEXT/FI.POLYF.FPD.10            ; FINISH UP
                                ;17161 =00
                                ;17162 FI.POLYD.FPD.10:
                                ;17163         ;00-----------------------------; ALL FLAGS CLEAR
                                ;17164         R[R3]_M[VA],                    ; SAVE VA
U 01F4, 0085,B592,4034,C047,00FE,2 ;17165      NEXT/IE.PACK.DONE               ; THATS ALL FOR NOW
                                ;17166
                                ;17167         ;01 ----------------------------; FAULT DETECTED BY FPA
                                ;17168         VA_M[VA]-ZLITO[8],              ; BUMP ADDRESS BACK
U 01F5, 0981,BC10,0030,44A7,001F,4 ;17169      NEXT/FI.POLYD.FPD.10            ;
                                ;17170
                                ;17171         ;10-----------------------------; SECCND HALF READ FAULT
                                ;17172         VA_M[VA]-ZLITO[4],              ; BUMP ADDRESS BACK
U 01F6, 0581,BC10,0030,24A7,001F,4 ;17173      NEXT/FI.POLYD.FPD.10            ;
                                ;17174
                                ;17175         ;11-----------------------------; IMPOSSIBLE BUT DO SOMETHING
                                ;17176         VA_M[VA]-ZLITO[8],              ; BUMP ADDRESS BACK
U 01F7, 0981,BC10,0030,44A7,001F,4 ;17177      NEXT/FI.POLYD.FPD.10            ;
```

B	1	MICRO2	1M(01)	J	5	MICRO2	1M(01)	E	10	MICRO2	1M(01)
C	1	MICRO2	1M(01)	K	5	MICRO2	1M(01)	F	10	MICRO2	1M(01)
D	1	MICRO2	1M(01)	L	5	MICRO2	1M(01)	G	10	MICRO2	1M(01)
E	1	MICRO2	1M(01	M	5	MICRO2	1M(01)	H	10	MICRO2	1M(01)
F	1	MICRO2	1M(01	N	5	MICRO2	1M(01)	I	10	MICRO2	1M(01)
G	1	MICRO2	1M(01,	B	6	MICRO2	1M(01)	J	10	MICRO2	1M(01)
H	1	MICRO2	1M(01)	C	6	MICRO2	1M(01)	K	10	MICRO2	1M(01)
I	1	MICRO2	1M(01)	D	6	MICRO2	1M(01)	L	10	MICRO2	1M(01)
J	1	MICRO2	1M(01)	E	6	MICRO2	1M(01)	M	10	MICRO2	1M(01)
K	1	MICRO2	1M(01)	F	6	MICRO2	1M(01)	N	10	MICRO2	1M(01)
L	1	MICRO2	1M(01)	G	6	MICRO2	1M(01)	B	11	MICRO2	1M(01)
M	1	MICRO2	1M(01)	H	6	MICRO2	1M(01)	C	11	MICRO2	1M(01)
N	1	MICRO2	1M(01)	I	6	MICRO2	1M(01)	D	11	MICRO2	1M(01)
B	2	MICRO2	1M(01)	J	6	MICRO2	1M(01)	E	11	MICRO2	1M(01)
C	2	MICRO2	1M(01,	K	6	MICRO2	1M(01)	F	11	MICRO2	1M(01)
D	2	MICRO2	1M(01)	L	6	MICRO2	1M(01)	G	11	MICRO2	1M(01)
E	2	MICRO2	1M(01)	M	6	MICRO2	1M(01)	H	11	MICRO2	1M(01)
F	2	MICRO2	1M(01)	N	6	MICRO2	1M(01)	I	11	MICRO2	1M(01)
G	2	MICRO2	1M(01)	B	7	MICRO2	1M(01)	J	11	MICRO2	1M(01)
H	2	MICRO2	1M(01)	C	7	MICRO2	1M(01)	K	11	MICRO2	1M(01)
I	2	MICRO2	1M(01)	D	7	MICRO2	1M(01)	L	11	MICRO2	1M(01)
J	2	MICRO2	1M(01)	E	7	MICRO2	1M(01)	M	11	MICRO2	1M(01)
K	2	MICRO2	1M(01)	F	7	MICRO2	1M(01)	N	11	MICRO2	1M(01)
L	2	MICRO2	1M(01)	G	7	MICRO2	1M(01)	B	12	MICRO2	1M(01)
M	2	MICRO2	1M(01)	H	7	MICRO2	1M(01)	C	12	MICRO2	1M(01)
N	2	MICRO2	1M(01)	I	7	MICRO2	1M(01)	D	12	MICRO2	1M(01)
B	3	MICRO2	1M(01)	J	7	MICRO2	1M(01)	E	12	MICRO2	1M(01)
C	3	MICRO2	1M(01)	K	7	MICRO2	1M(01)	F	12	MICRO2	1M(01)
D	3	MICRO2	1M(01)	L	7	MICRO2	1M(01)	G	12	MICRO2	1M(01)
E	3	MICRO2	1M(01)	M	7	MICRO2	1M(01)	H	12	MICRO2	1M(01)
F	3	MICRO2	1M(01)	N	7	MICRO2	1M(01)	I	12	MICRO2	1M(01)
G	3	MICRO2	1M(01)	B	8	MICRO2	1M(01)	J	12	MICRO2	1M(01)
H	3	MICRO2	1M(01)	C	8	MICRO2	1M(01)	K	12	MICRO2	1M(01)
I	3	MICRO2	1M(01)	D	8	MICRO2	1M(01)	L	12	MICRO2	1M(01)
J	3	MICRO2	1M(01)	E	8	MICRO2	1M(01)	M	12	MICRO2	1M(01)
K	3	MICRO2	1M(01)	F	8	MICRO2	1M(01)	N	12	MICRO2	1M(01)
L	3	MICRO2	1M(01)	G	8	MICRO2	1M(01)	B	13	MICRO2	1M(01)
M	3	MICRO2	1M(01)	H	8	MICRO2	1M(01)	C	13	MICRO2	1M(01)
N	3	MICRO2	1M(01)	I	8	MICRO2	1M(01)	D	13	MICRO2	1M(01)
B	4	MICRO2	1M(01,	J	8	MICRO2	1M(01)	E	13	MICRO2	1M(01)
C	4	MICRO2	1M(01)	K	8	MICRO2	1M(01)	F	13	MICRO2	1M(01)
D	4	MICRO2	1M(01)	L	8	MICRO2	1M(01)	G	13	MICRO2	1M(01)
E	4	MICRO2	1M(01)	M	8	MICRO2	1M(01)	H	13	MICRO2	1M(01)
F	4	MICRO2	1M(01)	N	8	MICRO2	1M(01)	I	13	MICRO2	1M(01)
G	4	MICRO2	1M(01)	B	9	MICRO2	1M(01)	J	13	MICRO2	1M(01)
H	4	MICRO2	1M(01)	C	9	MICRO2	1M(01)	K	13	MICRO2	1M(01)
I	4	MICRO2	1M(01)	D	9	MICRO2	1M(01)	L	13	MICRO2	1M(01)
J	4	MICRO2	1M(01)	E	9	MICRO2	1M(01)	M	13	MICRO2	1M(01)
K	4	MICRO2	1M(01)	F	9	MICRO2	1M(01)	N	13	MICRO2	1M(01)
L	4	MICRO2	1M(01)	G	9	MICRO2	1M(01)	B	14	MICRO2	1M(01)
M	4	MICRO2	1M(01)	H	9	MICRO2	1M(01)	C	14	MICRO2	1M(01)
N	4	MICRO2	1M(01)	I	9	MICRO2	1M(01)	D	14	MICRO2	1M(01)
B	5	MICRO2	1M(01)	J	9	MICRO2	1M(01)	E	14	MICRO2	1M(01)
C	5	MICRO2	1M(01)	K	9	MICRO2	1M(01)	F	14	MICRO2	1M(01)
D	5	MICRO2	1M(01)	L	9	MICRO2	1M(01)	G	14	MICRO2	1M(01)
E	5	MICRO2	1M(01)	M	9	MICRO2	1M(01)	H	14	MICRO2	1M(01)
F	5	MICRO2	1M(01)	N	9	MICRO2	1M(01)	I	14	MICRO2	1M(01)
G	5	MICRO2	1M(01)	B	10	MICRO2	1M(01)	J	14	MICRO2	1M(01)
H	5	MICRO2	1M(01)	C	10	MICRO2	1M(01)	K	14	MICRO2	1M(01)
I	5	MICRO2	1M(01)	D	10	MICRO2	1M(01)	L	14	MICRO2	1M(01)

M 14 MICRO2 1M(01)
N 14 MICRO2 1M(01)
B 15 MICRO2 1M(01)
C 15 MICRO2 1M(01)
D 15 MICRO2 1M(01)
E 15 MICRO2 1M(01)
F 15 MICRO2 1M(01)
G 15 MICRO2 1M(01)
H 15 MICRO2 1M(01)
I 15 MICRO2 1M(01)
J 15 MICRO2 1M(01)
K 15 MICRO2 1M(01)
L 15 MICRO2 1M(01)
M 15 MICRO2 1M(01)
N 15 MICRO2 1M(01)
B 16 MICRO2 1M(01)
C 16 MICRO2 1M(01)
D 16 MICRO2 1M(01)
E 16 MICRO2 1M(01)
F 16 MICRO2 1M(01)
G 16 MICRO2 1M(01)
H 16 MICRO2 1M(01)
I 16 MICRO2 1M(01)
J 16 MICRO2 1M(01)
K 16 MICRO2 1M(01)
L 16 MICRO2 1M(01)

```
                        ;17178 .TOC "  Floating point and CRC          :       FI.POLYF & FI.POLYD FPD RESTART ROUT
                        ;17179 ;*****************************************************************************
                        ;17180 ;
                        ;17181 ;
                        ;17182 ;       FPD RESTART ROUTINES
                        ;17183 ;
                        ;17184 ;               FI.POLYF.FPD.RES
                        ;17185 ;               FI.POLYD.FPD.RES
                        ;17186 ;
                        ;17187 ;
                        ;17188 ;
                        ;17189 ;*****************************************************************************
                        ;17190
                        ;17191 .REGION/IRD1.R1L,IRD1.R1H
                        ;17192 =000
                        ;17193 FI.POLYF.FPD.RES:                        ; ********************
                        ;17194         ;000-----------------------------; GARBAGE WRITE GET ARG
                        ;17195         FPA_MB M[TEMP1] R[R1],           ;
U 0240, 0886,15BE,4034,4047,1455,D      ;17196         PUSH,NEXT/FI.POLYFD.RES.10       ; RESET POINTERS
                        ;17197
                        ;17198         ;001-----------------------------; WRITE ARG GET PART PRODUCT
                        ;17199         FPA_MB M[TEMP1] R[R0],           ;
U 0241, 0486,15BE,4034,0047,1423,E      ;17200         PUSH,NEXT/FI.POLYF.OFF           ; SET FPDOFFSET
                        ;17201
                        ;17202         ;010-----------------------------;
                        ;17203         FPA_M[TEMP1],                    ; SEND PART PRODUCT
U 0242, 0080,1036,4030,0047,1023,C      ;17204         NEXT/FI.POLYF.50                 ; RETURN TO BUMP ITERATION COUNT
                        ;17205 =
                        ;17206
                        ;17207 =000
                        ;17208 FI.POLYD.FPD.RES:                        ; ********************
                        ;17209         ;000-----------------------------; WRITE GARBAGE GET ARG BIT 0-31
                        ;17210         FPA_MB M[TEMP1] R[R4],           ;
U 0258, 0886,15BE,4035,0047,1455,D      ;17211         PUSH,NEXT/FI.POLYFD.RES.10       ; RESET POINTERS
                        ;17212
                        ;17213         ;001-----------------------------; WRITE BITS 0-31 ARG GET BITS 32-63
                        ;17214         FPA_MB M[TEMP1] R[R5],           ;
U 0259, 0C86,15BE,4035,4047,1423,F      ;17215         PUSH,NEXT/FI.POLYD.OFF           ; SET FPDOFFSET
                        ;17216
                        ;17217         ;010-----------------------------; WRITE BITS 32-63 ARG
U 025A, 0486,15BE,4034,0047,1037,C      ;17218         FPA_MB M[TEMP1]_R[R0]            ; GET PART PROD BITS 0-31
                        ;17219 =
                        ;17220         ;-------------------------------; WRITE PART PRODUCT BITS 0-31
U 037C, 0886,15BE,4034,4047,1037,D      ;17221         FPA_MB M[TEMP1]_R[R1]            ; GET PART PROD BITS 32-63
                        ;17222
                        ;17223         ;-------------------------------;
                        ;17224         FPA_M[TEMP1],                    ; WRITE PART PRODUCT BITS 32-63
U 037D, 0880,1036,4030,0047,1037,E      ;17225         NEXT/FI.POLYD.71                 ; RETURN TO BUMP ITERATION COUNT
                        ;17226                                          ; FIXED, POST CMT062
```

```
                                ;17227 .TOC "  Floating point and CRC          :       FI.POLYD    "
                                ;17228 .REGION/IRD1.R1L,IRD1.R1H
                                ;17229 =0
                                ;17230 FI.POLYD:                                ; **********************
                                ;17231          ;0------------------------------;
                                ;17232          R[TEMP3] M[MDR],                ; BITS 32-63 OF ARG SAVED
                                ;17233          PUSH,FLAG<1-0>?,                ; BITS 0-31 IN TEMP1
U 0252, 0485,2592,4530,C047,0421,4 ;17234       NEXT/FI.POLYD.150               ; OR TEMP0 OR SHORT LIT
                                ;17235
                                ;17236          ;1------------------------------;
                                ;17237          WB M[TEMP4].ANDNOT.ZLIT0[1F],   ; DEGREE VALID
U 0253, 0900,4C13,8A70,F847,0020,4 ;17238       CLEAR FLAG0,WX.NE.0?            ;
                                ;17239
                                ;17240 =00
                                ;17241          ,00-----------------------------; DEGREE VALID
                                ;17242          R[TEMP4].SIZ RB-1,              ; DECREMENT FOR OTHER READ
                                ;17243          SIZE[BYTE],MDR 0,               ; MDR 0 FOR DEG =0 CASE
                                ;17244          CLEAR READ(FLAG1),              ; CLEAR FLAG1 FOR SECOND HALF READ
                                ;17245          PUSH,SIGND CMP?,                ; WHAT IS DEGREE SIZE
U 0204, 040A,0E7D,0B41,04E7,0C26,0 407* ;17246  NEXT/FI.POLYD.30                 ;
                                ;17247
                                ;17248          ;01-----------------------------; DEGREE > 31
U 0205, 0C80,003',4030,0047,00FF,8 ;17249       NEXT/IE.OPER.FAULT              ;
                                ;17250
                                ;17251 =11
                                ;17252          ;11-----------------------------; RET +3 IF DEG =0,1
                                ;17253          R[R1] FPA,                      ; WRITE BITS 32-63 OF ANSWER
U 0207, 0084,0036,4034,4047,4020,A ;17254       NEXT/FI.POLYD.55                ; FINISH UP
                                ;17255
                                ;17256 =000
                                ;17257 FI.POLYD.30:
                                ;17258          ;000----------------------------; DEGREE >1
                                ;17259          SET READ(FLAG1),                ;
                                ;17260          READ,VA_VA+4,SIZE[LONG],        ; FINISH UP READ SEND TO FPA AND
U 0260, 0848,0036,4020,0450,047E,6 ;17261       PUSH,NEXT/FI.POLYFD.RW1         ; GET FIRST PART OF ANSWER
                                ;17262
                                ;17263          ;001----------------------------; DEGREE =1
                                ;17264          SET READ(FLAG1),                ;
                                ;17265          READ,VA_VA+4,SIZE[LONG],        ; FINISH UP READ,SEND GET ANS
U 0261, 0048,0036,4020,0450,007E,6 ;17266       NEXT/FI.POLYFD.RW1              ;
                                ;17267
                                ;17268          ;010----------------------------; DEGREE = 0
                                ;17269          FPA M[MDR],                     ; SEND A 0 TO NEGATE ARG
U 0262, 0C81,2036,4030,0047,1037,7 ;17270       NEXT/FI.POLYFD.RW2              ; FINISH UP READ,SEND GET ANS
                                ;17271
                                ;17272
                                ;17273          ;011----------------------------; RET+3 IF DEGREE > 1
                                ;17274          R[R4] M[TEMP2],PUSH,            ; SAVE FIRST HALF OF ARG
U 0263, 0084,2592,4035,0047,047F,C ;17275       NEXT/FI.POLYFD.SET              ; SET FPD,GPRS
                                ;17276
                                ;17277          ;100----------------------------;
                                ;17278          R[R5] M[TEMP3],                 ; SAVE BITS 32-63 OF ARG
U 0264, 0084,3592,4035,4047,0020,B ;17279       NEXT/FI.POLYD.70                ; START NEXT ROUND
                                ;17280 =
```

```
                                ;17281 =00
                                ;17282 FI.POLYD.50:
                                ;17283          ;00-------------------------------; LOOP > 0 NO INT OR TIMER
                                ;17284          READ,VA_VA+4,SIZE[LONG],        ; GET 1ST PART COEF
                                ;17285          FPAWAIT,SET READ(FLAG1),        ;
U 0208, 0A48,0036,4020,0450,0423,A ;17286       PUSH,NEXT/FI.POLYFD.RW3.5       ; CONTINUE ON
                                ;17287
                                ;17288          ;01-------------------------------; LOOP > 0 INT OR TIMER
                                ;17289          PUSH,INTPEND OR TIMER?,         ; RET -1 IF TIMER ONLY
U 0209, 0480,0036,4AF0,0047,04F7,0 ;17290       NEXT/IE.SERV.IP.TS2             ;
                                ;17291
                                ;17292 FI.POLYD.55:
                                ;17293          ;10-------------------------------; LOOP < 0 DONE
                                ;17294          R[R4]_0,                        ; ZERO OUT GPRS
U 020A, 0C84,05B7,0035,0047,0070,F ;17295       NEXT/FI.POLYD.220               ; FINISH UP
                                ;17296
                                ;17297 FI.POLYD.70:
                                ;17298          ;11-------------------------------; RET +3 FROM LOOP > 0 READ
U 020B, 0884,0036,4034,4047,4037,E ;17299       R[R1]_FPA                       ; SAVE IT IN GPR
                                ;17300
                                ;17301 FI.POLYD.71:                             ; LABEL ADDED FOR CMT094
                                ;17302          ;-------------------------------;
                                ;17303          R[R2].SIZ_RB-1,SIZE[BYTE],      ; DECREMENT ITERATION COUNT
                                ;17304          CLEAR READ(FLAG1),              ; CLEAR 2ND HALF READ FLAG
                                ;17305          COUNT OR  INT TIMER?,           ; LOOP COMPLETE OR TIMER TIME
U 037E, 0C0A,0E7D,0B04,8047,0820,8 410* ;17306  NEXT/FI.POLYD.50                 ; GO SEE
                                ;17307
```

```
                                ;17308 =00
                                ;17309 FI.POLYD.150:
                                ;17310          ;00-------------------------------; NOT REG OR LITERAL
                                ;17311          R[TEMP2] M[TEMP1],                ; ARG BITS 0-31 IN TEMP1
                                ;17312          LOD INC BRA?,                     ; GET DEGREE RETURN TO FI.POLYFD.10
U 0214, 0C84,1592,41B0,8047,0010,0  ;17313      NEXT/OS.RED                       ; THROUGH IRDROM
                                ;17314
                                ;17315          ;01-------------------------------; REG NOT LITERAL
                                ;17316          R[TEMP2] M[TEMP0],                ; ARG BITS 0-31 IN TEMP0
                                ;17317          LOD INC BRA?,                     ; GET DEGREE RETURN TO FI.POLYFD.10
U 0215, 0884,0592,41B0,8047,0010,0  ;17318      NEXT/OS.RED                       ; THROUGH IRDROM
                                ;17319
                                ;17320          ;10-------------------------------; LITERAL MODE
                                ;17321          R[TEMP2] M[TEMP3].FPLIT,          ; UNPACK SHORT LITERAL
                                ;17322          CLEAR FLAG1,                      ; CLEAR FLAG
U 0216, 0C0C,37B7,0030,8047,0037,F  ;17323      NEXT/FI.POLYD.155                 ; CONTINUE ON
                                ;17324
                                ;17325          ;11-------------------------------; LITERAL MODE BUT IMPOSSIBLE
                                ;17326          R[TEMP2] M[TEMP3].FPLIT,          ; UNPACK SHORT LITERAL
                                ;17327          CLEAR FLAG1,                      ; CLEAR FLAG
U 0217, 0C0C,37B7,0030,8047,0037,F  ;17328      NEXT/FI.POLYD.155                 ; CONTINUE ON
                                ;17329
                                ;17330 FI.POLYD.155:
                                ;17331          ;-------------------------------;
                                ;17332          R[TEMP3] 0,                       ; ZERO OUT SECOND HALF
                                ;17333          CLEAR FLAG0,                      ; BE SURE FLAG IS CLEAR
                                ;17334          LOD INC BRA?,                     ; GET DEGREE
U 037F, 0404,05B7,01B0,C047,0010,0  ;17335      NEXT/OS.RED                       ; RET THRU IRDROM TO FI.POLYFD.10
```

```
                        ;17336 .TOC    ''       Floating point and CRC          : CRC''
                        ;17337
                        ;17338 ;*****************************************************************************
                        ;17339 ;
                        ;17340 ;       OB      CRC     tbl.ab,initialcrc.rl,stream.rw,stream.ab [R0-R3.wl]
                        ;17341 ;
                        ;17342 ;               Input:
                        ;17343 ;                       MDR             tbl.ab
                        ;17344 ;
                        ;17345 ;               Resources:
                        ;17346 ;                       Temp0,Temp1     temporary CRC
                        ;17347 ;                       Temp2           table index calculation
                        ;17348 ;                       Temp3           temporary table address
                        ;17349 ;                       Temp4           temporary for # bytes (string length)
                        ;17350 ;                       Temp7           PC during instruction
                        ;17351 ;
                        ;17352 ;                       FPDOFFSET       temporary for FPD index
                        ;17353 ;                       FPD
                        ;17354 ;                       PC              used to prefetch source byte
                        ;17355 ;                       Q               CRC after XOR with source byte
                        ;17356 ;                                       during inner loop iteration
                        ;17357 ;                       Flag1           for fault/interrupt differentiation
                        ;17358 ;                       Platch
                        ;17359 ;                       Stepcounter
                        ;17360 ;
                        ;17361 ;                               GPRs during instruction
                        ;17362 ;
                        ;17363 ;                       R0              CRC
                        ;17364 ;                       R1              table address
                        ;17365 ;                       R2              byte 0-1 length of string
                        ;17366 ;                                       byte 2   PCdelta if ever an FPD pack
                        ;17367 ;                                       byte 3 (bit 31) set if table page fault
                        ;17368 ;                       R3              stream address
                        ;17369 ;
                        ;17370 ;       Output:
                        ;17371 ;                       R0              CRC
                        ;17372 ;                       R1,R2           0
                        ;17373 ;                       R3              stream address +1
                        ;17374 ;
                        ;17375 ;       Subroutines:
                        ;17376 ;                       OS.RED          get initial crc
                        ;17377 ;                                       get stream length
                        ;17378 ;                       OS.ADD          get stream address
                        ;17379 ;                       IE.SERV.IP.TS2  process interrupt or timer
                        ;17380 ;
                        ;17381 ;       Operation:
                        ;17382 ;                       Start up
                        ;17383 ;                       S1.     Get all operands
                        ;17384 ;                       S2.     Set FPD
                        ;17385 ;                       S3.     Set platch and FPDOFFSET to 3
                        ;17386 ;                       S4.     Save PC in temp7
                        ;17387 ;                       S5.     Save stream address in R3 and PC
                        ;17388 ;                       S6.     Save length in R2
                        ;17389 ;                       S7.     If length is 0  - done
                        ;17390 ;                               Otherwise main processing
```

```
                                                    Main processing
                                                    M1.     #bytes >0 and no interrupt or timer pending
                                                            Get source byte from XB
                                                    M2.     XOR byte with CRC in R0 and save in Q
                                                            and in Temp0
                                                    M3.     Enter inner loop

                                                    Inner loop proceessing
                                                    I1.     Get bits 0-3 of CRC  if non zero
                                                    I2.     Multiply by 8 and use as index into table
                                                    I3.     Get the long word table entry
                                                    I4.     XOR entry with CRC
                                                            (which has been shifted the 4 bits right)
                                                    I5.     Save CRC in R0
                                                    I6.     Repeat a second time from I1.

                                                    Finish up
                                                    F1.     Clear R2
                                                    F2.     R3 gets PC
                                                    F3.     PC gets Temp7
                                                    F4.     Clear R1 and IRD1

                                                    If interrupt/fault occurs
                                                    FPD.PACK
                                                    P1.     Save PC in R3 (stream address)
                                                    P2.     Restore PC from Temp7
                                                    P3.     Calculate PC delta from PC and PCBACK
                                                    P4.     If page fault while reading table
                                                            (flag 1 set) then set bit 31 of R2
                                                            also save the value of the CRC in Q
                                                            effectively roll back to start of inner loop

                                                    FPD.RESTART
                                                    R1.     Compute Pc value for Temp7 from
                                                            PCBACK and PCdelta
                                                    R2.     Set FPDOFFSET and Platch
                                                    R3.     PC gets stream address from R3
                                                    R4.     If bit 31 of R2 not set
                                                            get stream byte again from XB
                                                            and continue processing
                                                            If bit 31 set  then set Flag1
                                                            clear bit 31
                                                            Restore Q and Temp0 from R0
                                                            set stepcounter to 2 and
                                                            continue at inner loop
                                                    ;*******************************************************************
```

```
                                                                H  1
                                ;17438 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;17439 CR.CRC:                          ; ***********
                                ;17440          ;-----------------------------;
                                ;17441          R[TEMP3]_M[MDR],        ; SAVE TABLE ADDRESS
                                ;17442          LOD INC BRA?,           ; GET INITIAL CRC
U 0381, 0C85,2592,41B0,C047,0010,0  ;17443      NEXT/OS.RED             ; RET THRU IRDROM TO CR.CRC.10
                                ;17444
                                ;17445 =00
                                ;17446 CR.CRC.10:
                                ;17447          ;00-------------------------;
                                ;17448          R[TEMP1]_M[MDR],        ; SAVE CRC
                                ;17449          PUSH,                   ; GET SOURCE STREAM LENGTH
                                ;17450          LOD INC BRA?,           ; RETURN +1
U 0224, 0085,2592,41B0,4047,0410,0  ;17451      NEXT/OS.RED
                                ;17452
                                ;17453          ;01-------------------------;
                                ;17454          R[TEMP4]_ZEXT(M[MDR]),  ; ZERO EXTEND THE # BYTES
                                ;17455          SIZE[WORD],             ;
                                ;17456          LOD INC BRA?,           ; GET THE STREAM ADDRESS
U 0225, 0C85,259E,4191,0047,0412,0  ;17457      PUSH,NEXT/OS.ADD        ; RETURN +1
                                ;17458
                                ;17459          ;10-------------------------;
                                ;17460          R[R1]_M[TEMP3],         ; SAVE THE TABLE ADDRESS
                                ;17461          SET FPD,                ; FIRST PART DONE
                                ;17462          PUSH,                   ; SET PLATCH AND FPDOFFSET
U 0226, 04EC,3592,4034,4047,0471,9  ;17463      NEXT/CR.SETPLOFF
                                ;17464
                                ;17465          ;11-------------------------;
U 0227, 0485,A592,4031,C047,0071,0  ;17466      R[TEMP7]_M[PC]          ; SAVE THE PC
                                ;17467
                                ;17468 .REGION/FLOAT.R1L,FLOAT.R1H/FLOAT.R2L,FLOAT.R2H/FLOAT.R3L,FLOAT.R3H
                                ;17469          ;-----------------------------;
U 0710, 0885,2592,4034,C487,0072,0  ;17470      PC_R[R3]_M[MDR]         ; STREAM ADDRESS TO PC AND SAVE IT
                                ;17471
                                ;17472 0720:    ;*********FORCE ADDRESS*********;
                                ;17473          ;-----------------------------;
U 0720, 0D80,0C37,0030,00A7,0071,1  ;17474      CC_[0]                  ; CLEAR CONDTION CODES TO START.
                                ;17475
                                ;17476          ;-----------------------------;
                                ;17477          R[R0]_M[TEMP1],SIZE[LONG],  ; SAVE CRC AND
U 0711, 0C84,1592,4024,1047,0071,2  ;17478      CCOP2                   ; SET CONDITION CODES
                                ;17479
                                ;17480          ;-----------------------------;
                                ;17481          R[R2]_M[TEMP4],         ; SAVE STREAM LENGTH IN R2
U 0712, 0484,4592,4A74,8047,006D,E  ;17482      WX.NE.0?                ; IS THE LENGTH 0
                                ;17483 =0
                                ;17484 CR.DONE2:
                                ;17485          ;0--------------------------; YES, DONE
                                ;17486          PC_R[TEMP7],            ; RESTORE THE PC
                                ;17487          CLEAR FPD,              ; CLEAR THE FPD BIT
U 06DE, 00E0,05BE,4031,C487,0071,4  ;17488      NEXT/CR.DONE3           ; JOIN COMMON END UP
```

```
                                                                ;17489 CR.CONT:
                                                                ;17490          ;1------------------------------;  MAIN PROCESSING LOOP
                                                                ;17491          R[R2].SIZ_RB-1,                 ;  DECREMENT # BYTES
                                                                ;17492          SIZE[LONG],                     ;
                                                                ;17493          CLEAR FLAG1,                    ;  BE SURE FLAG IS CLEAR IF FAULT
U 06DF, 000A,0E7D.0B24,8047,0866,4 410*                         ;17494          COUNT OR INT TIMER?             ;  TEST FOR DONE AND/OR INTERRUPT/TIMER
                                                                ;17495
                                                                ;17496 =00
                                                                ;17497 CR.CONT1:
                                                                ;17498          ;00-----------------------------;  LOOP >=0 NO INTERRUPT OR TIMER
                                                                ;17499          R[TEMP0]_ZEXT(X8) PC_PC+1,      ;  PREFETCH SOURCE BYTE
U 0664, 0C85,759E,4000,2047,0071,5                              ;17500          NEXT/CR.CONT2                   ;
                                                                ;17501
                                                                ;17502          ;01-----------------------------;  LOOP >=0 BUT INTERRUPT OR TIMER
                                                                ;17503          INTPEND OR TIMER?,              ;  IF ONLY TIMER RETURN -1
U 0665, 0480,0036,4AF0,0047,04F7,0                              ;17504          PUSH,NEXT/IE.SERV.IP.TS2        ;  MORE PROCESSING
                                                                ;17505
                                                                ;17506          ;10-----------------------------;  LOOP <0 DONE
                                                                ;17507          R[R3]_M[PC],                    ;  SAVE STREAM ADDRESS
U 0666, 04E5,A592,4034,C047,0071,3                              ;17508          CLEAR FPD                       ;  CLEAR THE FPD BIT
                                                                ;17509 =
                                                                ;17510          ;-------------------------------;
                                                                ;17511          R[R2]_0,                        ;  ZERO OUT THE GPR
U 0713, 0484,05B7,0034,8047,006D,E                              ;17512          NEXT/CR.DONE2                   ;
                                                                ;17513
                                                                ;17514 CR.DONE3:
                                                                ;17515          ;-------------------------------;
                                                                ;17516          R[R1]_0,                        ;  ZERO OUT OTHER TEMP
U 0714, 0084,05B7,0134,4047,003F,9                              ;17517          IRD1                            ;  END OF INSTRUCTION
                                                                ;17518
                                                                ;17519 CR.CONT2:
                                                                ;17520          ;-------------------------------;
                                                                ;17521          M[TEMP0]_Q_MB.XOR.R[R0],        ;  XOR CRC AND SOURCE BYTE
U 0715, 04A6,0003,5034,0047,0066,8                              ;17522          STEPC_2                         ;  SET INNER LOOP CONTROL TO 2
                                                                ;17523
                                                                ;17524 =00
                                                                ;17525 CR.INNER:
                                                                ;17526          ;00-----------------------------;
                                                                ;17527          R[TEMP2]_M[TEMP0].NIBBLE,       ;  GET THE LOW ORDER 4 BITS
                                                                ;17528          MDR_0,                          ;  SET MDR TO 0 FOR LATER USE
                                                                ;17529          SET FLAG1,                      ;  SET FLAG IN CASE OF READ TABLE FAULT
                                                                ;17530          WX.NE.0?,                       ;  WERE THE 4 BITS 0
U 0668, 084C,03F7,0A70,84E7,0466,C                              ;17531          PUSH,NEXT/CR.GETTBL             ;  DO THE INNER TABLE PROCESSING
                                                                ;17532
                                                                ;17533          ;01-----------------------------;  END OF INNER LOOP
                                                                ;17534          R[R0]_M[TEMP0],                 ;  SET THE CRC
                                                                ;17535          CCOP2,SIZE[LONG],               ;  ALSO THE CONDITION CODES
; 0669, 0884,0592,4024,1047,006D,F                               ;17536          NEXT/CR.CONT                    ;  NEXT ITERATION
                                                                ;17537
                                                                ;17538          ;10-----------------------------;  RET +2 FROM GETTBL
                                                                ;17539          R[TEMP0]_M[MDR].XOR.RB,         ;  XOR CRC AND THE TABLE ENTRY
                                                                ;17540          DBZ STEPC?,                     ;  STEPCOUNTER 0?
U 066A, 0885,2003,4330,0047,0066,8                              ;17541          NEXT/CR.INNER
                                                                ;17542 =
```

```
                                        ;17543 =00
                                        ;17544 CR.GETTBL:
                                        ;17545          ;00------------------------------; INDEX MAY BE 0
                                        ;17546          R[TEMP0]_(M[MDR] RB).RR.4,      ; SHIFT THE CRC RIGHT 4 BITS
U 066C, 0C85,2277,00B0,0047,0000,2      ;17547          RETURN [+2]
                                        ;17548
                                        ;17549          ;01------------------------------;
                                        ;17550          VA_R[R1]+(M[TEMP2].ASL.P),       ; MULT INDEX
                                        ;17551          PUSH,                           ; ADD TO TABLE ADDRESS AND
U 066D, 0C80,2A7D,0034,44A7,0466,C      ;17552          NEXT/CR.GETTBL                  ; STORE IN VA NOW SHIFT CRC
                                        ;17553 =11
                                        ;17554          ;11------------------------------; RET +2 FROM SHIFT CRC
                                        ;17555          READ,                           ; READ TABLE ENTRY
                                        ;17556          SIZE[LONG],                     ; INTO MDR
U 066F, 0C80,0036,40A0,0050,0000,2      ;17557          RETURN [+2]                     ; RETURN TO  INNER LOOP
                                        ;17558
                                        ;17559 0719:    ;*********FORCE ADDRESS*********;
                                        ;17560 CR.SETPLOFF.
                                        ;17561          ;-------------------------------;
U 0719, 0D86,CC37,0030,1847,0071,6      ;17562          M[FPDOFFSET]_ZLIT0[3]           ; SET FPDOFFSET
                                        ;17563
                                        ;17564          ;-------------------------------;
                                        ;17565          PL_[2],                         ; SET PLATCH FOR THE INDEX SHIFT
U 0716, 0180,0EF6,40B0,1047,0000,1      ;17566          RETURN [+1]
                                        ;17567
                                        ;17568 ;
                                        ;17569 ;        THIS INSTRUCTION AT FPDOFFSET #3 FOR CRC
                                        ;17570 ;
                                        ;17571 ;        ;-------------------------------;
                                        ;17572 ;        R[R3]_M[PC],                    ; SAVE THE STREAM ADDRESS
                                        ;17573 ;        NEXT/CR.CRC.SAVE
                                        ;17574 ;
                                        ;17575
                                        ;17576 CR.CRC.SAVE:
                                        ;17577          ;-------------------------------;
U 0717, 0818,05BE,4031,C487,0071,8      ;17578          PC_R[TEMP7],CLEAR FLAG3         ; RESTORE  PC CLEAR FLAG3 - IE.PACK.DONE
                                        ;17579
                                        ;17580          ;-------------------------------;
                                        ;17581          R[TEMP7]_RB-M[PCBACK],          ; CALCULATE PC DELTA
U 0718, 0C85,9003,05F1,C047,0062,0      ;17582          FLAG1?                          ; WAS THIS TABLE READ FAULT
                                        ;17583
                                        ;17584 =0*
                                        ;17585 CR.CRC.SAVE.10:
                                        ;17586          ;0*------------------------------; NO
                                        ;17587          R[R2]_RB.OR.(M[TEMP7].RR.SIZ),   ; MERGE PC DELTA AND BYTE COUNT
                                        ;17588          SIZE[WORD],                     ;
U 0620, 0084,73BE,4014,8047,00FE,2      ;17589          NEXT/IE.PACK.DONE               ;
                                        ;17590
                                        ;17591          ;1*------------------------------; YES TABLE READ FAULT
U 0622, 0D86,7D52,4030,4047,0071,A      ;17592          M[TEMP7]_MB.OR.ZLIT12[8]        ; SET THE BIT 31  FOR  RESTART
                                        ;17593
                                        ;17594          ;-------------------------------;
                                        ;17595          R[R0]_Q,Q_D,                    ; SAVE Q CLOBBERS IT
U 071A, 0884,002C,7034,0047,0062,0      ;17596          NEXT/CR.CRC.SAVE.10             ; D IS GARBAGE
```

```
;22274 .TOC     ''      Character String                : Operand fetch and Initialization''
;22275
;22276          ;*****************************************************************************
;22277          ;       MOVC3
;22278          ;
;22279          ;       OUTPUT          R0              LENGTH
;22280          ;                       D                  ''
;22281          ;                       Q                  ''
;22282          ;                       TEMP1
;22283          ;                       R1              SOURCE ADDRESS
;22284          ;                       PC                 ''      ''
;22285          ;                       TEMP2              ''      ''
;22286          ;                       R3              DESTINATION ADDRESS
;22287          ;                       VA                 ''      ''
;22288          ;                       R2              DELTA PC'0(8BITS'24BITS)
;22289          ;                       TEMP6           1(ROTATED LEFT BY 8)
;22290          ;                       CC SET          (R0-R0)
;22291          ;                       FPD             SET
;22292          ;
;22293          ;       RESOURCES       TEMP6           FOR SAVING DELTA PC
;22294          ;
;22295          ;       SUBROUTINES     CS.EXP.FILLER   EXPANDS TEMP5
;22296          ;                       CS.R2.GETS.DELPC.R2   R2_DELTA PC'R2
;22297          ;*****************************************************************************
;22298
;22299          ;*****************************************************************************
;22300          ;       MOVC5
;22301          ;
;22302          ;       OUTPUT          R0              MIN(SRC LGTH,DEST LGTH)
;22303          ;                       Q                  ''      ''      ''
;22304          ;                       D                  ''      ''      ''
;22305          ;                       R1              SOURCE ADDRESS
;22306          ;                       PC                 ''      ''
;22307          ;                       TEMP2              ''      ''
;22308          ;                       TEMP5           FILL(EXPANDED)
;22309          ;                       TEMP3           DESTINATION LENGTH
;22310          ;                       FLAG1           SET(FOR CMPC5 AND MOVC5)
;22311          ;                       MTEMP10         SRC LGTH - DST LGTH
;22312          ;                       R3              DESTINATION ADDRESS
;22313          ;                       VA                 ''      ''
;22314          ;                       R2              DELTA PC'0(8BITS'24BITS)
;22315          ;                       TEMP6           1(ROTATED LEFT BY 8)
;22316          ;                       CC SET          (TEMP10_R0-TMP3)
;22317          ;                       FPD             SET
;22318          ;
;22319          ;       RESOURCES       TEMP6           FOR SAVING DELTA PC
;22320          ;
;22321          ;       SUBROUTINES     CS.EXP.FILLER   EXPANDS TEMP5
;22322          ;                       CS.R2.GETS.DELPC.R2   R2_DELTA PC'R2
;22323          ;*****************************************************************************
```

```
;24378 .TOC     "       Character String                : MATCHC"
;24379
;24380          ;*****************************************************************************
;24381          ;       MATCHC
;24382          ;       INPUT           R2              DELTA PC'0'OBJECT LENGTH
;24383          ;                                       (8 BITS)'(8)'(16 BITS)
;24384          ;                       Q
;24385          ;                       R3              OBJECT ADDRESS
;24386          ;                       VA
;24387          ;                       R0              SOURCE LENGTH
;24388          ;                       D
;24389          ;                       PC              SOURCE ADDRESS
;24390          ;                       R1
;24391          ;                       FLAG3           SET (MATCHC FLAG)
;24392          ;
;24393          ;       OUTPUT          R0              #OF UNMAT BYTES OBJ STR
;24394          ;                       R1              ADD NEXT BYTE IN OBJ STR
;24395          ;                                       OR ONE BYTE BEYOND
;24396          ;                       R2              # BYTES LEFT IN SRC STR
;24397          ;                       R3              ADD OF NEXT BYTE BEYOND
;24398          ;                                       MATCH OR SRC STR
;24399          ;                       CC              NZVC_0100 IF R0=0
;24400          ;                                       NZVC_0000 ELSE
;24401          ;
;24402          ;       RESOURCES       R0
;24403          ;                       R1
;24404          ;                       R2
;24405          ;                       R3
;24406          ;                       FPDOFFSET       7
;24407          ;                       VA
;24408          ;                       PC
;24409          ;                       TEMP1
;24410          ;                       TEMP6           CODE
;24411          ;                       Q
;24412          ;                       D
;24413          ;                       MTMP10
;24414          ;                       FLAG0
;24415          ;                       FLAG3           SET FOR MATCHC
;24416          ;
;24417          ;       SUBROUTINES     CS.GEN.RESPC
;24418          ;*****************************************************************************
```

```
;17696  .ICODE  ;      OPS          REG                    MEM                OPS       FPA REG                FPA MEM
;17697       051: FPD [NOP][IE.OPCOD.DEC ]                                    [NOP][IE.OPCOD.DEC      ],                  ;CMPF
;17698            IRD1[LOD][OS.FRED      ]                                    [LOD][OS.RED            ]
;17699  .OCODE
;17700       051: CNT0[LOD][OS.FRED      ]LOS.FRED          ]  [LOD][FI.CMPF.REG    ][OS.RED          ],
;17701            CNT1[NOP][FP.CMPF      ][FP.CMPF          ]  [NOP][FI.CMPF.MEM    ][FI.CMPF.MEM     ]
;17702
;17703  .ICODE
;17704       00B: FPD [NOP][CR.CRC.RES   ]                                    [NOP][CR.CRC.RES        ],                  ;CRC
;17705            IRD1[LOD][OS.ADD       ]                                    [LOD][OS.ADD            ]
;17706  .OCODE
;17707       00B: CNT0[NOP][CR.CRC       ][CR.CRC           ]  [NOP][CR.CRC         ][CR.CRC          ],
;17708            CNT1[NOP][CR.CRC.10    ][CR.CRC.10        ]  [NOP][CR.CRC.10      ][CR.CRC.10       ]
;17709
;17710  .ICODE
;17711       06C: FPD [NOP][IE.OPCOD.DEC ]                                    [NOP][IE.OPCOD.DEC      ],                  ;CVTBD
;17712            IRD1[LOD][OS.RED       ]                                    [LOD][OS.RED            ]
;17713  .OCODE
;17714       06C: CNT0[NOP][FP.CVT.BWL.FD][FP.CVT.BWL.FD    ]  [NOP][FI.ADDD3.MEM   ][FI.ADDD3.MEM    ],
;17715            CNT1[NOP][FP.CVT.BWL.FD.100][FP.CVT.BWL.FD.100]  [NOP][FI.ADDD3.20  ][FI.ADDD3.20   ]
;17716
;17717  .ICODE
;17718       04C: FPD [NOP][IE.OPCOD.DEC ]                                    [NOP][IE.OPCOD.DEC      ],                  ;CVTBF
;17719            IRD1[LOD][OS.RED       ]                                    [LOD][OS.RED            ]
;17720  .OCODE
;17721       04C: CNT0[NOP][FP.CVT.BWL.FD][FP.CVT.BWL.FD    ]  [NOP][FI.ADDF3.MEM   ][FI.ADDF3.MEM    ],
;17722            CNT1[NOP][FP.CVT.BWL.FD.100][FP.CVT.BWL.FD.100]  [NOP][FI.ADDF3.20  ][FI.ADDF3.20   ]
;17723
;17724  .ICODE
;17725       068: FPD [NOP][IE.OPCOD.DEC ]                                    [NOP][IE.OPCOD.DEC      ],                  ;CVTDB
;17726            IRD1[LOD][OS.DRED      ]                                    [LOD][OS.FIDRED         ]
;17727  .OCODE
;17728       068: CNT0[NOP][FP.CVTD.BWL  ][FP.CVTD.BWL      ]  [NOP][FI.ADDF3.MEM   ][FI.ADDF3.MEM    ],
;17729            CNT1[NOP][FP.CVTD.150  ][FP.CVTD.150      ]  [NOP][FI.ADDF3.20  ][FI.ADDF3.20   ]
;17730
;17731  .ICODE
;17732       076: FPD [NOP][IE.OPCOD.DEC ]                                    [NOP][IE.OPCOD.DEC      ],                  ;CVTDF
;17733            IRD1[LOD][OS.DRED      ]                                    [LOD][OS.FIDRED         ]
;17734  .OCODE
;17735       076: CNT0[LOD][OS.WRT2      ][OS.WRT2          ]  [NOP][FI.CVTDFD      ][FI.CVTDFD       ],
;17736            CNT1[NOP][FP.CVTDF     ][FP.CVTDF         ]  [NOP][FI.CVTDF.10    ][FI.CVTDF.10     ]
;17737
;17738  .ICODE
;17739       06A: FPD [NOP][IE.OPCOD.DEC ]                                    [NOP][IE.OPCOD.DEC      ],                  ;CVTDL
;17740            IRD1[LOD][OS.DRED      ]                                    [LOD][OS.FIDRED         ]
;17741  .OCODE
;17742       06A: CNT0[NOP][FP.CVTD.BWL  ][FP.CVTD.BWL      ]  [NOP][FI.ADDF3.MEM   ][FI.ADDF3.MEM    ],
;17743            CNT1[NOP][FP.CVTD.150  ][FP.CVTD.150      ]  [NOP][FI.ADDF3.20  ][FI.ADDF3.20   ]
;17744
;17745  .ICODE
;17746       069: FPD [NOP][IE.OPCOD.DEC ]                                    [NOP][IE.OPCOD.DEC      ],                  ;CVTDW
;17747            IRD1[LOD][OS.DRED      ]                                    [LOD][OS.FIDRED         ]
;17748  .OCODE
;17749       069: CNT0[NOP][FP.CVTD.BWL  ][FP.CVTD.BWL      ]  [NOP][FI.ADDF3.MEM   ][FI.ADDF3.MEM    ],
;17750            CNT1[NOP][FP.CVTD.150  ][FP.CVTD.150      ]  [NOP][FI.ADDF3.20  ][FI.ADDF3.20   ]
```

```
;17751  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;17752       048: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;CVTFB
;17753            IRD1[LOD][OS.FRED           ]                          [LOD][OS.RED            ]
;17754  .OCODE
;17755       048: CNT0[LOD][OS.WRT2           ][OS.WRT2           ]  [NOP][FI.ADDF3.MEM      ][FI.ADDF3.MEM      ],
;17756            CNT1[NOP][FP.CVTF.BWL       ][FP.CVTF.BWL       ]  [NOP][FI.ADDF3.20       ][FI.ADDF3.20       ]
;17757
;17758  .ICODE
;17759       056: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;CVTFD
;17760            IRD1[LOD][OS.FRED           ]                          [LOD][OS.RED            ]
;17761  .OCODE
;17762       056: CNT0[LOD][FP.CVTFD          ][FP.CVTFD          ]  [NOP][FI.CVTDFD         ][FI.CVTDFD         ],
;17763            CNT1[NOP][FP.CVTFD.30       ][FP.CVTFD.30       ]  [NOP][FI.CVTFD.10       ][FI.CVTFD.10       ]
;17764
;17765  .ICODE
;17766       04A: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;CVTFL
;17767            IRD1[LOD][OS.FRED           ]                          [LOD][OS.RED            ]
;17768  .OCODE
;17769       04A: CNT0[LOD][OS.WRT2           ][OS.WRT2           ]  [NOP][FI.ADDF3.MEM      ][FI.ADDF3.MEM      ],
;17770            CNT1[NOP][FP.CVTF.BWL       ][FP.CVTF.BWL       ]  [NOP][FI.ADDF3.20       ][FI.ADDF3.20       ]
;17771
;17772  .ICODE
;17773       049: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;CVTFW
;17774            IRD1[LOD][OS.FRED           ]                          [LOD][OS.RED            ]
;17775  .OCODE
;17776       049: CNT0[LOD][OS.WRT2           ][OS.WRT2           ]  [NOP][FI.ADDF3.MEM      ][FI.ADDF3.MEM      ],
;17777            CNT1[NOP][FP.CVTF.BWL       ][FP.CVTF.BWL       ]  [NOP][FI.ADDF3.20       ][FI.ADDF3.20       ]
;17778
;17779  .ICODE
;17780       06E: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;CVTLD
;17781            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;17782  .OCODE
;17783       06E: CNT0[NOP][FP.CVT.BWL.FD     ][FP.CVT.BWL.FD     ]  [NOP][FI.ADDD3.MEM      ][FI.ADDD3.MEM      ],
;17784            CNT1[NOP][FP.CVT.BWL.FD.100 ][FP.CVT.BWL.FD.100 ]  [NOP][FI.ADDD3.20       ][FI.ADDD3.20       ]
;17785
;17786  .ICODE
;17787       04E: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;CVTLF
;17788            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;17789  .OCODE
;17790       04E: CNT0[LOD][OS.WRT2           ][OS.WRT2           ]  [NOP][FI.ADDF3.MEM      ][FI.ADDF3.MEM      ],
;17791            CNT1[NOP][FP.CVTLF          ][FP.CVTLF          ]  [NOP][FI.ADDF3.20       ][FI.ADDF3.20       ]
;17792
;17793  .ICODE
;17794       06B: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;CVTRDL
;17795            IRD1[LOD][OS.DRED           ]                          [LOD][OS.FIDRED         ]
;17796  .OCODE
;17797       06B: CNT0[NOP][FP.CVTRDL         ][FP.CVTRDL         ]  [NOP][FI.ADDF3.MEM      ][FI.ADDF3.MEM      ],
;17798            CNT1[NOP][FP.CVTD.150       ][FP.CVTD.150       ]  [NOP][FI.ADDF3.20       ][FI.ADDF3.20       ]
;17799
;17800  .ICODE
;17801       04B: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;CVTRFL
;17802            IRD1[LOD][OS.FRED           ]                          [LOD][OS.RED            ]
;17803  .OCODE
;17804       04B: CNT0[LOD][OS.WRT2           ][OS.WRT2           ]  [NOP][FI.ADDF3.MEM      ][FI.ADDF3.MEM      ],
;17805            CNT1[NOP][FP.CVTRFL         ][FP.CVTRFL         ]  [NOP][FI.ADDF3.20       ][FI.ADDF3.20       ]
```

```
.ICODE  ;       OPS             REG                     MEM                     OPS             FPA REG                 FPA MEM
        06D: FPD [NOP][IE.OPCOD.DEC         ]                               [NOP][IE.OPCOD.DEC              ],                      ;CVTWD
             IRD1[LOD][OS.RED               ]                               [LOD][OS.RED                    ]
.OCODE
        06D: CNT0[NOP][FP.CVT.BWL.FD        ][FP.CVT.BWL.FD             ]   [NOP][FI.ADDD3.MEM              ][FI.ADDD3.MEM          ],
             CNT1[NOP][FP.CVT.BWL.FD.100    ][FP.CVT.BWL.FD.100         ]   [NOP][FI.ADDD3.20               ][FI.ADDD3.20           ]

.ICODE
        04D: FPD [NOP][IE.OPCOD.DEC         ]                               [NOP][IE.OPCOD.DEC              ],                      ;CVTWF
             IRD1[LOD][OS.RED               ]                               [LOD][OS.RED                    ]
.OCODE
        04D: CNT0[NOP][FP.CVT.BWL.FD        ][FP.CVT.BWL.FD             ]   [NOP][FI.ADDF3.MEM              ][FI.ADDF3.MEM          ],
             CNT1[NOP][FP.CVT.BWL.FD.100    ][FP.CVT.BWL.FD.100         ]   [NOP][FI.ADDF3.20               ][FI.ADDF3.20           ]

.ICODE
        066: FPD [NOP][IE.OPCOD.DEC         ]                               [NOP][IE.OPCOD.DEC              ],                      ;DIVD2
             IRD1[LOD][OS.DRED              ]                               [LOD][OS.FIDRED                 ]
.OCODE
        066: CNT0[NOP][FP.DIVD2             ][FP.DIVD2                  ]   [LOD][FI.ADDD2.REG              ][OS.FIDMOD             ],
             CNT1[NOP][FP.DIVD.20           ][FP.DIVD.20                ]   [NOP][FI.ADDD2.MEM              ][FI.ADDD2.MEM          ]

.ICODE
        067: FPD [NOP][IE.OPCOD.DEC         ]                               [NOP][IE.OPCOD.DEC              ],                      ;DIVD3
             IRD1[LOD][OS.DRED              ]                               [LOD][OS.FIDRED                 ]
.OCODE
        067: CNT0[NOP][FP.DIVD3             ][FP.DIVD3                  ]   [LOD][FI.ADDD3.REG              ][OS.FIDRED             ],
             CNT1[NOP][FP.DIVD.20           ][FP.DIVD.20                ]   [NOP][FI.ADDD3.MEM              ][FI.ADDD3.MEM          ]

.ICODE
        046: FPD [NOP][IE.OPCOD.DEC         ]                               [NOP][IE.OPCOD.DEC              ],                      ;DIVF2
             IRD1[LOD][OS.FRED              ]                               [LOD][OS.RED                    ]
.OCODE
        046: CNT0[LOD][OS.MOD               ][OS.MOD                    ]   [LOD][FI.ADDF2.REG              ][OS.MOD                ],
             CNT1[NOP][FP.DIVF23            ][FP.DIVF23                 ]   [NOP][FI.ADDF2.MEM              ][FI.ADDF2.MEM          ]

.ICODE
        047: FPD [NOP][IE.OPCOD.DEC         ]                               [NOP][IE.OPCOD.DEC              ],                      ;DIVF3
             IRD1[LOD][OS.FRED              ]                               [LOD][OS.RED                    ]
.OCODE
        047: CNT0[LOD][OS.FRED              ][OS FRED                   ]   [LOD][FI.ADDF3.REG              ][OS.RED                ],
             CNT1[NOP][FP.DIVF23            ][FP.DIVF23                 ]   [NOP][FI.ADDF3.MEM              ][FI.ADDF3.MEM          ]

.ICODE
        074: FPD [NOP][IE.OPCOD.DEC         ]                               [NOP][IE.OPCOD.DEC              ],                      ;EMODD
             IRD1[LOD][OS.DRED              ]                               [LOD][OS.FIDRED                 ]
.OCODE
        074: CNT0[NOP][FP.EMODD             ][FP.EMODD                  ]   [LOD][OS.RED                    ][OS.RED                ],
             CNT1[NOP][FP.EMODD.10          ][FP.EMODD.10               ]   [NOP][FI.EMODD                  ][FI.EMODD              ]

.ICODF
        054: FPD [NOP][IE.OPCOD.DEC         ]                               [NOP][IE.OPCOD.DEC              ],                      ;EMODF
             IRD1[LOD][OS.FRED              ]                               [LOD][OS.RED                    ]
.OCODE
        054: CNT0[LOD][OS.RED               ][OS.RED                    ]   [LOD][OS.RED                    ][OS.RED                ],
             CNT1[NOP][FP.EMODF             ][FP.EMODF                  ]   [NOP][FI.EMODF                  ][FI.EMODF              ]
```

```
;17861  .ICODE  ;       OPS          REG                    MEM              OPS        FPA REG              FPA MEM
;17862        072: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],              ;MNEGD
;17863             IRD1[LOD][OS.DRED             ]                        [LOD][OS.DRED             ]
;17864  .OCODE
;17865        072: CNT0[LOD][FP.MNEGD.REG        ][OS.WRT2          ]  [LOD][FP.MNEGD.REG        ][OS.WRT2          ],
;17866             CNT1[NOP][FP.MNEGD.MEM        ][FP.MNEGD.MEM     ]  [NOP][FP.MNEGD.MEM        ][FP.MNEGD.MEM     ]
;17867
;17868  .ICODE
;17869        052: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],              ;MNEGF
;17870             IRD1[LOD][OS.FRED             ]                        [LOD][OS.FRFD             ]
;17871  .OCODE
;17872        052: CNT0[LOD][FP.MNEGF.REG        ][OS.WRT2          ]  [LOD][FP.MNEGF.REG        ][OS.WRT2          ],
;17873             CNT1[NOP][FP.MNEGF.MEM        ][FP.MNEGF.MEM     ]  [NOP][FP.MNEGF.MEM        ][FP.MNEGF.MEM     ]
;17874
;17875  ;SEE MOVAQ                                                                                                  MOVAD
;17876
;17877  ;SEE MOVAL                                                                                                  MOVAF
;17878
;17879  .ICODE
;17880        070: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],              ;MOVD
;17881             IRD1[LOD][OS.DRED             ]                        [LOD][OS.DRED             ]
;17882  .OCODE
;17883        070: CNT0[LOD][FP. 'OVD.REG        ][OS.WRT2          ]  [LOD][FP.MOVD.REG         ][OS.WRT2          ],
;17884             CNT1[NOP][FP.MOVD.MEM         ][FP.MOVD.MEM      ]  [NOP][FP.MOVD.MEM         ][FP.MOVD.MEM      ]
;17885
;17886  .ICODE
;17887        050: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],              ;MOVF
;17888             IRD1[LOD][OS.FRED             ]                        [LOD][OS.FRED             ]
;17889  .OCODE
;17890        050: CNT0[LOD][FP.MOVF.REG         ][OS.WRT2          ]  [LOD][FP.MOVF.REG         ][OS.WRT2          ],
;17891             CNT1[NOP][FP.MOVF.MEM         ][FP.MOVF.MEM      ]  [NOP][FP.MOVF.MEM         ][FP.MOVF.MEM      ]
;17892
;17893  .ICODE
;17894        064: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],              ;MULD2
;17895             IRD1[LOD][OS.DRED             ]                        [LOD][OS.FIDRED           ]
;17896  .OCODE
;17897        064: CNT0[NOP][FP.MULD2            ][FP.MULD2         ]  [LOD][FI.ADDD2.REG        ][OS.FIDMOD        ],
;17898             CNT1[NOP][FP.MULD.20          ][FP.MULD.20       ]  [NOP][FI.ADDD2.MEM        ][FI.ADDD2.MEM     ]
;17899
;17900  .ICODE
;17901        065: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],              ;MULD3
;17902             IRD1[LOD][OS.DRED             ]                        [LOD][OS.FIDRED           ]
;17903  .OCODE
;17904        065: CNT0[NOP][FP.MULD3            ][FP.MULD3         ]  [LOD][FI.ADDD3.REG        ][OS.FIDRED        ],
;17905             CNT1[NOP][FP.MULD.20          ][FP.MULD.20       ]  [NOP][FI.ADDD3.MEM        ][FI.ADDD3.MEM     ]
;17906
;17907  .ICODE
;17908        044: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],              ;MULF2
;17909             IRD1[LOD][OS.FRED             ]                        [LOD][OS.RED              ]
;17910  .OCODE
;17911        044: CNT0[LOD][OS.MOD              ][OS.MOD           ]  [LOD][FI.ADDF2.REG        ][OS.MOD           ],
;17912             CNT1[NOP][FP.MULF23           ][FP.MULF23        ]  [NOP][FI.ADDF2.MEM        ][FI.ADDF2.MEM     ]
```

```
;17913  .ICODE  ;       OPS          REG                    MEM              OPS        FPA REG              FPA MEM
;17914       045: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;MULF3
;17915            IRD1[LOD][OS.FRED           ]                          [LOD][OS.RED            ]
;17916  .OCODE
;17917       045: CNT0[LOD][OS.FRED           ][OS.FRED          ]  [LOD][FI.ADDF3.REG    ][OS.RED          ],
;17918            CNT1[NOP][FP.MULF23         ][FP.MULF23        ]  [NOP][FI.ADDF3.MEM    ][FI.ADDF3.MEM    ]
;17919
;17920  .ICODE
;17921       075: FPD [NOP][FP.POLYD.FPD.RES  ]                          [NOP][FI.POLYD.FPD.RES  ],                  ;POLYD
;17922            IRD1[LOD][OS.DRED           ]                          [LOD][OS.FIDRED         ]
;17923  .OCODE
;17924       075: CNT0[NOP][FP.POLYD          ][FP.POLYD         ]  [NOP][FI.POLYD        ][FI.POLYD        ],
;17925            CNT1[NOP][FP.POLYD.10       ][FP.POLYD.10      ]  [NOP][FI.POLYFD.10    ][FI.POLYFD.10    ]
;17926
;17927  .ICODE
;17928       055: FPD [NOP][FP.POLYF.FPD.RES  ]                          [NOP][FI.POLYF.FPD.RES  ],                  ;POLYF
;17929            IRD1[LOD][OS.FRED           ]                          [LOD][OS.FRED           ]
;17930  .OCODE
;17931       055: CNT0[LOD][OS.RED            ][OS.RED           ]  [LOD][OS.RED          ][OS.RED          ],
;17932            CNT1[NOP][FP.POLYF          ][FP.POLYF         ]  [NOP][FI.POLYF        ][FI.POLYF        ]
;17933
;17934  ;SEE PUSHAQ                                                                                              PUSHAD
;17935
;17936  ;SEE PUSHAL                                                                                              PUSHAF
;17937
;17938  .ICODE
;17939       062: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;SUBD2
;17940            IRD1[LOD][OS.DRED           ]                          [LOD][OS.FIDRED         ]
;17941  .OCODE
;17942       062: CNT0[NOP][FP.SUBD2          ][FP.SUBD2         ]  [LOD][FI.ADDD2.REG    ][OS.FIDMOD       ],
;17943            CNT1[NOP][FP.ADDD.20        ][FP.ADDD.20       ]  [NOP][FI.ADDD2.MEM    ][FI.ADDD2.MEM    ]
;17944
;17945  .ICODE
;17946       063: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;SUBD3
;17947            IRD1[LOD][OS.DRED           ]                          [LOD][OS.FIDRED         ]
;17948  .OCODE
;17949       063: CNT0[NOP][FP.SUBD3          ][FP.SUBD3         ]  [LOD][FI.ADDD3.REG    ][OS.FIDRED       ],
;17950            CNT1[NOP][FP.ADDD.20        ][FP.ADDD.20       ]  [NOP][FI.ADDD3.MEM    ][FI.ADDD3.MEM    ]
;17951
;17952  .ICODE
;17953       042: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;SUBF2
;17954            IRD1[LOD][OS.FRED           ]                          [LOD][OS.RED            ]
;17955  .OCODE
;17956       042: CNT0[NOP][FP.SUBF2          ][FP.SUBF2         ]  [LOD][FI.ADDF2.REG    ][OS.MOD          ],
;17957            CNT1[NOP][FP.SUBF23.20      ][FP.SUBF23.20     ]  [NOP][FI.ADDF2.MEM    ][FI.ADDF2.MEM    ]
;17958
;17959  .ICODE
;17960       043: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;SUBF3
;17961            IRD1[LOD][OS.FRED           ]                          [LOD][OS.RED            ]
;17962  .OCODE
;17963       043: CNT0[NOP][FP.SUBF3          ][FP.SUBF3         ]  [LOD][FI.ADDF3.REG    ][OS.RED          ],
;17964            CNT1[NOP][FP.SUBF23.20      ][FP.SUBF23.20     ]  [NOP][FI.ADDF3.MEM    ][FI.ADDF3.MEM    ]
```

```
;17965  .ICODE  ;      OPS          REG                    MEM            OPS        FPA REG                FPA MEM
;17966        073: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ],                    ;TSTD
;17967             IRD1[LOD][OS.DRED           ]                        [LOD][OS.DRED           ]
;17968  .OCODE
;17969        073: CNT0[NOP][FP.TSTD           ][FP.TSTD           ]  [NOP][FP.TSTD           ][FP.TSTD           ],
;17970             CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;17971
;17972  .ICODE
;17973        053: FPD [NOP][IE.OPCOD.DEC      ]                        [NOP][IE.OPCOD.DEC      ],                    ;TSTF
;17974             IRD1[LOD][OS.FRED           ]                        [LOD][OS.FRED           ]
;17975  .OCODE
;17976        053: CNT0[NOP][FP.TSTF           ][FP.TSTF           ]  [NOP][FP.TSTF           ][FP.TSTF           ],
;17977             CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;17978
```

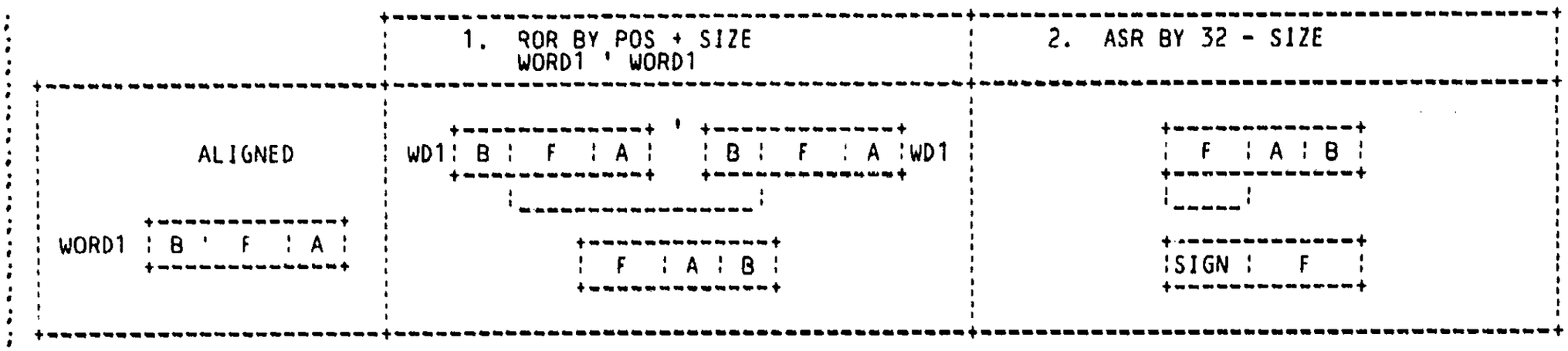
```
;17979  .TOC "  Floating point and CRC          : Dsize Rom Definition"
;17980  .DCODE
;17981
;17982  06F:    SIZE [DBLE] [DBLE] [DBLE] [   0] [   0] [   0]  ;ACBD
;17983
;17984  04F:    SIZE [FLOT] [FLOT] [FLOT] [   0] [   0] [   0]  ;ACBF
;17985
;17986  060:    SIZE [DBLE] [DBLE] [   0] [   0] [   0] [   0]  ;ADDD2
;17987
;17988  061:    SIZE [DBLE] [DBLE] [DBLE] [   0] [   0] [   0]  ;ADDD3
;17989
;17990  040:    SIZE [FLOT] [FLOT] [   0] [   0] [   0] [   0]  ;ADDF2
;17991
;17992  041:    SIZE [FLOT] [FLOT] [FLOT] [   0] [   0] [   0]  ;ADDF3
;17993
;17994  ;SEE CLRQ                                                ;CLRD
;17995
;17996  ;SEE CLRL                                                ;CLRF
;17997
;17998  071:    SIZE [DBLE] [DBLE] [   0] [   0] [   0] [   0]  ;CMPD
;17999
;18000  051:    SIZE [FLOT] [FLOT] [   0] [   0] [   0] [   0]  ;CMPF
;18001
;18002  00B:    SIZE [BYTE] [LONG] [WORD] [BYTE] [   0] [   0]  ;CRC
;18003
;18004  06C:    SIZE [BYTE] [DBLE] [   0] [   0] [   0] [   0]  ;CVTBD
;18005
;18006  04C:    SIZE [BYTE] [FLOT] [   0] [   0] [   0] [   0]  ;CVTBF
;18007
;18008  068:    SIZE [DBLE] [BYTE] [   0] [   0] [   0] [   0]  ;CVTDB
;18009
;18010  076:    SIZE [DBLE] [FLOT] [   0] [   0] [   0] [   0]  ;CVTDF
;18011
;18012  06A:    SIZE [DBLE] [LONG] [   0] [   0] [   0] [   0]  ;CVTDL
;18013
;18014  069:    SIZE [DBLE] [WORD] [   0] [   0] [   0] [   0]  ;CVTDW
;18015
;18016  048:    SIZE [FLOT] [BYTE] [   0] [   0] [   0] [   0]  ;CVTFB
;18017
;18018  056:    SIZE [FLOT] [DBLE] [   0] [   0] [   0] [   0]  ;CVTFD
;18019
;18020  04A:    SIZE [FLOT] [LONG] [   0] [   0] [   0] [   0]  ;CVTFL
;18021
;18022  049:    SIZE [FLOT] [WORD] [   0] [   0] [   0] [   0]  ;CVTFW
;18023
;18024  06E:    SIZE [LONG] [DBLE] [   0] [   0] [   0] [   0]  ;CVTLD
;18025
;18026  04E:    SIZE [LONG] [FLOT] [   0] [   0] [   0] [   0]  ;CVTLF
;18027
;18028  06B:    SIZE [DBLE] [LONG] [   0] [   0] [   0] [   0]  ;CVTRDL
;18029
;18030  04B:    SIZE [FLOT] [LONG] [   0] [   0] [   0] [   0]  ;CVTRFL
;18031
;18032  06D:    SIZE [WORD] [DBLE] [   0] [   0] [   0] [   0]  ;CVTWD
```

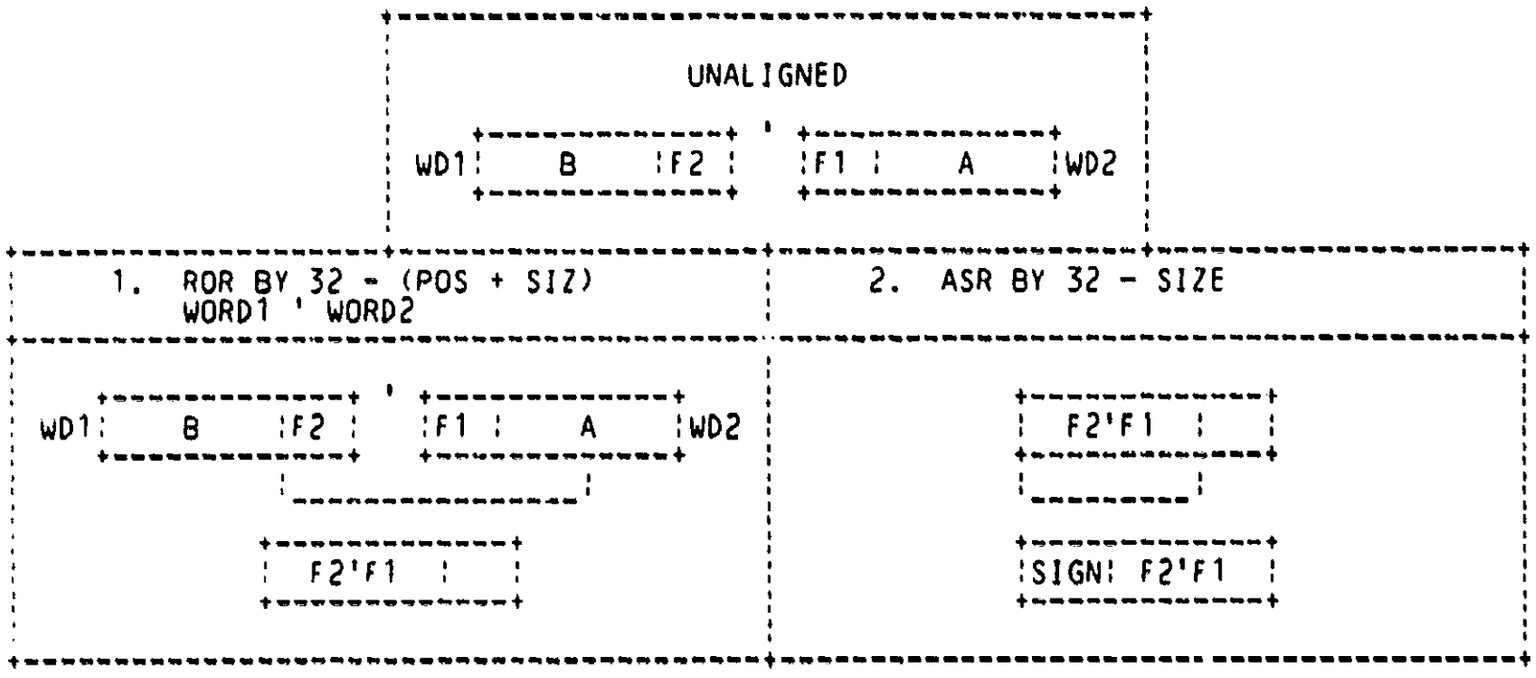
```
;18033  04D:    SIZE [WORD] [FLOT] [   0] [   0] [   0] [   0]  ;CVTWF
;18034
;18035  066:    SIZE [DBLE] [DBLE] [   0] [   0] [   0] [   0]  ;DIVD2
;18036
;18037  067:    SIZE [DBLE] [DBLE] [DBLE] [   0] [   0] [   0]  ;DIVD3
;18038
;18039  046     SIZE [FLOT] [FLOT] [   0] [   0] [   0] [   0]  ;DIVF2
;18040
;18041  047:    SIZE [FLOT] [FLOT] [FLOT] [   0] [   0] [   0]  ;DIVF3
;18042
;18043  074:    SIZE [DBLE] [BYTE] [DBLE] [LONG] [DBLE] [   0]  ;EMODD
;18044
;18045  054:    SIZE [FLOT] [BYTE] [FLOT] [LONG] [FLOT] [   0]  ;EMODF
;18046
;18047  072:    SIZE [DBLE] [DBLE] [   0] [   0] [   0] [   0]  ;MNEGD
;18048
;18049  052:    SIZE [FLOT] [FLOT] [   0] [   0] [   0] [   0]  ;MNEGF
;18050
;18051  ;SEE MOVAQ                                               ;MOVAD
;18052
;18053  ;SEE MOVAL                                               ;MOVAF
;18054
;18055  070:    SIZE [DBLE] [DBLE] [   0] [   0] [   0] [   0]  ;MOVD
;18056
;18057  050:    SIZE [FLOT] [FLOT] [   0] [   0] [   0] [   0]  ;MOVF
;18058
;18059  064:    SIZE [DBLE] [DBLE] [   0] [   0] [   0] [   0]  ;MULD2
;18060
;18061  065:    SIZE [DBLE] [DBLE] [DBLE] [   0] [   0] [   0]  ;MULD3
;18062
;18063  044:    SIZE [FLOT] [FLOT] [   0] [   0] [   0] [   0]  ;MULF2
;18064
;18065  045:    SIZE [FLOT] [FLOT] [FLOT] [   0] [   0] [   0]  ;MULF3
;18066
;18067  075:    SIZE [DBLE] [WORD] [BYTE] [   0] [   0] [   0]  ;POLYD
;18068
;18069  055:    SIZE [FLOT] [WORD] [BYTE] [   0] [   0] [   0]  ;POLYF
;18070
;18071  ;SEE PUSHAQ                                              ;PUSHAD
;18072
;18073  ;SEE PUSHAL                                              ;PUSHAF
;18074
;18075  062:    SIZE [DBLE] [DBLE] [   0] [   0] [   0] [   0]  ;SUBD2
;18076
;18077  063:    SIZE [DBLE] [DBLE] [DBLE] [   0] [   0] [   0]  ;SUBD3
;18078
;18079  042:    SIZE [FLOT] [FLOT] [   0] [   0] [   0] [   0]  ;SUBF2
;18080
;18081  043:    SIZE [FLOT] [FLOT] [FLOT] [   0] [   0] [   0]  ;SUBF3
;18082
;18083  073:    SIZE [DBLE] [   0] [   0] [   0] [   0] [   0]  ;TSTD
;18084
;18085  053:    SIZE [FLOT] [   0] [   0] [   0] [   0] [   0]  ;TSTF
;18086  .UCODE
                                        ;18087 .BIN
```

18087; This page intentionally left blank.

```
                                        ;18088 .TOC "VIELD.MIC"
                                        ;18089 .TOC "REVISION 7.0"
                                        ;18090 ;     G. Koeckhoven, P. R. GUILBAULT
                                        ;18091
;18092  .NOBIN
;18093  .TOC    "       Revision History"
;18094
;18095  ; 07    force some address.
;18096  ; 06    Initial release.
                                        ;18097          .BIN
```

```
                                ;18098 .TOC"  Variable Length Bit Field      : Algorithm For EXTV & CMPV"
                                ;18099
;18100  .NOBIN
```





```
;18146  .TOC"   Variable Length Bit Field       : Algorithm For EXTZV, CMPZV, FFS, FFC"
;18147
;18148
;18149
;18150
;18151
;18152  ;  +-----------------------------------------+                +-----------------------------------------+
;18153  ;  |                                         |                |                                         |
;18154  ;  |                ALIGNED                  |                |                UNALIGNED                |
;18155  ;  |                                         |                |                                         |
;18156  ;  |                 +-------------+         |                |  +-------------+ ' +-------------+      |
;18157  ;  |          WORD1  | B | F | A   |         |                |  |WD1| B   |F2 | | |F1 | A   |WD2|      |
;18158  ;  |                 +-------------+         |                |  +-------------+ ' +-------------+      |
;18159  ;  |                                         |                |                                         |
;18160  ;  +-----------------------------------------+----------------+-----------------------------------------+
;18161  ;  |                                         |                |                                         |
;18162  ;  |  +-------------+ ' +-------------+      |  EXTRACT & ZERO EXT.  |  +-------------+ ' +-------------+      |
;18163  ;  |WD1| B | F | A |   | B | F | A |WD1|      |  WORD1 ' WORD1        |  |WD1| B   |F2 | | |F1 | A   |   |      |
;18164  ;  |  +-------------+   +-------------+      |                |  +-------------+ ' +-------------+      |
;18165  ;  |                      |_____|            |                |                |_________|              |
;18166  ;  |                                         |                |                                         |
;18167  ;  |                 +-------------+         |                |              +-------------+            |
;18168  ;  |                 |  0  |  F    |         |                |              | 0 | F2'F1   |            |
;18169  ;  |                 +-------------+         |                |              +-------------+            |
;18170  ;  |                                         |                |                                         |
;18171  ;  +-----------------------------------------+----------------+-----------------------------------------+
                                ;18172 .BIN
```

```
                                ;18173 .TOC     "       Variable Length Bit Field       : FFS, FFC"
                                ;18174
                                ;18175 ;*****************************************************************************
                                ;18176 ;       FF(S,C)   startpos.rl size.rb base.ab findpos.wl
                                ;18177 ;
                                ;18178 ;           Input        OSR             Base
                                ;18179 ;                        MDR             Size
                                ;18180 ;                        Q               Startpos
                                ;18181 ;
                                ;18182 ;           Resources    TEMP1           Startpos
                                ;18183 ;                        TEMP2           Extracted Field & Work
                                ;18184 ;                        TEMP3           Work
                                ;18185 ;                        SLATCH,D        Size
                                ;18186 ;                        FLAG<2:0>       001
                                ;18187 ;                        FLAG3           Size = 32
                                ;18188 ;
                                ;18189 ;           Subroutines  VF.VSUB
                                ;18190 ;*****************************************************************************
                                ;18191
                                ;18192 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;18193 =00000
                                ;18194 =01111
                                ;18195 VF.FFS:
                                ;18196          ;01111-------------------------; EA
                                ;18197          PUSH,SET FLAG0,SIZE[IDEP],      ;
                                ;18198          SL_D_SEXT(M[MDR]) WBRANGE?,     ; SL <- D <- SIZE & FLAG<2:0> <- 1
U 00EF, 0041,2B56,6DFD,8047,0D7E,8 324*         ;18199          NEXT7VF.VSUB                    ; BUT ON RANGE OF SIZE TO VIELD SUB
                                ;18200
                                ;18201          ;10000-------------------------;
U 00F0, 0086,35BF,0030,8047,017E,4              ;18202          M[TEMP3]_-R[TEMP2]              ; TEMP3 <- -(EXTRACTED FIELD)
                                ;18203 =
                                ;18204
                                ;18205 .REGION/VIELD.R1L,VIELD.R1H/VIELD.R2L,VIELD.R2H/VIELD.R3L,VIELD.R3H
                                ;18206          ;-------------------------------;
                                ;18207          M[TEMP3]_MB.AND.R[TEMP2],       ; TEMP3 <- 1 IN BIT POS OF 1ST BIT SET
U 17E4, 0086,3002,0030,8047,0026,7              ;18208          NEXT/VF.FFS.OR.FFCSIZ32         ;
                                ;18209
                                ;18210
                                ;18211 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;18212 =00000
                                ;18213 =01111
                                ;18214 VF.FFC:
                                ;18215          ;01111-------------------------; EB
                                ;18216          PUSH,SET FLAG0,SIZE[IDEP],      ;
                                ;18217          SL_D_SEXT(M[MDR]) WBRANGE?,     ; SL <- D <- SIZE & FLAG<2:0> <- 1
U 01EF, 0041,2B56,6DFD,8047,0D7E,8 324*         ;18218          NEXT7VF.VSUB                    ; BUT ON RANGE OF SIZE TO VIELD SUB
                                ;18219
                                ;18220          ;10000-------------------------;
U 01F0, 0C84,2E50,0030,C047,017E,6              ;18221          R[TEMP3]_M[TEMP2]+1             ; TEMP3 <- (EXTRACTED FILED)+1
                                ;18222 =
                                ;18223
                                ;18224 .REGION/VIELD.R1L,VIELD.R1H/VIELD.R2L,VIELD.R2H/VIELD.R3L,VIELD.R3H
                                ;18225          ;-------------------------------;
                                ;18226          M[TEMP3]_MB.ANDNOT.R[TEMP2],    ; TEMP3 <- 1 IN BIT POS OF 1ST BIT CLEAR
U 17E6, 0486,3003,84F0,84E7,0026,6              ;18227          MDR_0,FLAG3?                    ; SIZE.EQ.32?
```

```
                                        ;18228 =0
                                        ;18229          ;0------------------------------; FFC SIZE.EQ.[0,31]
                                        ;18230          WB M[TEMP3] PL MB.MSS,          ; PL <- POS OF 1ST BIT CLEAR (PL WILL
U 0266, 0080,39C2,403D,8047,0038,2      ;18231          NEXT/VF.FFC.SIZ0TO31            ; POINT 1 PAST FIELD IF BIT NOT FOUND)
                                        ;18232
                                        ;18233 VF.FFS.OR.FFCSIZ32:
                                        ;18234          ;1------------------------------; (FFC SIZE.EQ.32) OR FFS
                                        ;18235          WB M[TEMP3] PL MB.MSS,CCOP1,    ; PL <- POS OF 1ST BIT SET/CLEAR
U 0267, 0C80,39C2,4A3D,8847,0026,C      ;18236          SIZE[IDEP],WX.EQ.0?             ; BIT FOUND?
                                        ;18237 =0
                                        ;18238 VF.FFSFOUND.FFC0TO31:
                                        ;18239          ;0------------------------------; WX.NE.0 : BIT FOUND
                                        ;18240          R[DST.R] M[TEMP1]+PL,SIZE[IDEP],; WRITE FIND POSITION
U 026C, 0C84,1B11,013C,45DA,003F,9      ;18241          WRITE NOTREG,IRD1               ;
                                        ;18242
                                        ;18243          ;1------------------------------; WX.EQ.0 : BIT NOT FOUND
                                        ;18244          R[DST.R] M[TEMP1]+SL,SIZE[IDEP],; WRITE FIND POSITION
U 026D, 0884,1B91,013C,45DA,003F,9      ;18245          WRITE NOTREG,IRD1               ;
                                        ;18246
                                        ;18247 VF.FFC.SIZ0TO31:
                                        ;18248          ;------------------------------;
                                        ;18249          WB D-PL,SIZE[IDEP],CCOP1,       ; IF SIZE = FINDPOS THEN BIT NOT FOUND
U 0382, 0880,0B30,0030,0847,0026,C      ;18250          NEXT/VF.FFSFOUND.FFC0TO31       ;
```

```
                                ;18251 .TOC     "       Variable Length Bit Field       : EXTV, EXTZV"
                                ;18252
                                ;18253 ;*****************************************************************************
                                ;18254 ;       EXT(V,ZV) pos.rl size.rb base.ab dst.wl
                                ;18255 ;
                                ;18256 ;           Input       OSR             Base
                                ;18257 ;                       MDR             Size
                                ;18258 ;                       Q               Pos
                                ;18259 ;
                                ;18260 ;           Resources   TEMP2           Extracted Field & Work
                                ;18261 ;                       PLATCH          -Size(EXTV)
                                ;18262 ;                       SLATCH,D        Size
                                ;18263 ;                       FLAG<2:0>       EXTV=000, EXTZV=001
                                ;18264 ;
                                ;18265 ;           Subroutines VF.VSUB
                                ;18266 ;*****************************************************************************
                                ;18267
                                ;18268 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;18269 =00000
                                ;18270 =01111
                                ;18271 VF.EXTZV:
                                ;18272          ;01111----------------------------; EF
                                ;18273          PUSH,SET FLAG0,SIZE[IDEP],      ;
                                ;18274          SL_D_SEXT(M[MDR]) WBRANGE?,     ; SL <- D <- SIZE & FLAG<2:0> <- 1
U 020F, 0041,2B56,6DFD,8047,0D7E,8 324*         ;18275          NEXT7VF.VSUB                    ; BUT ON RANGE OF SIZE TO VIELD SUB
                                ;18276
                                ;18277          ;10000----------------------------;
                                ;18278          R[DST.R]_M[TEMP2],WRITE NOTREG, ;
U 0210, 0C84,2592,413C,55DA,003F,9              ;18279          SIZE[IDEP],CCOP2,IRD1           ; WRITE DEST
                                ;18280 =
                                ;18281
                                ;18282
                                ;18283 22F:
                                ;18284 VF.EXTV:
                                ;18285          ;01111****FORCE ADDRESS*********; EE
                                ;18286          PUSH,SL_D_SEXT(M[MDR]) WBRANGE?,; SL <- D <- SIZE & FLAG<2:0> <- 0
U 022F, 0081,2B56,6DFD,8047,0D7E,8 324*         ;18287          SIZE[IDEP],NEXT/VF.VSUB         ; BUT ON RANGE OF SIZE TO VIELD SUB
                                ;18288
                                ;18289 230:     ;10000****FORCE ADDRESS*********;
                                ;18290          R[DST.R]_M[TEMP2],ASR.P,CCOP2,  ;
U 0230, 0884,20F7,013C,55DA,003F,9              ;18291          WRITE NOTREG,SIZE[IDEP],IRD1    ; WRITE DEST
```

```
                                ;18292 .TOC     ''      Variable Length Bit Field       : CMPV, CMPZV''
                                ;18293
                                ;18294 ;*************************************************************************
                                ;18295 ;       CMPZ(V,ZV) pos.rl size.rb base.ab src.rl
                                ;18296 ;
                                ;18297 ;          Input        OSR             Base
                                ;18298 ;                       MDR             Size
                                ;18299 ;                       Q               Pos
                                ;18300 ;
                                ;18301 ;          Resources    TEMP2           Extracted field & Work
                                ;18302 ;                       PLATCH          -Size(CMPV)
                                ;18303 ;                       SLATCH,D        Size
                                ;18304 ;                       FLAG<2:0>       CMPV=100, CMPZV=010
                                ;18305 ;
                                ;18306 ;          Subroutines  VF.VSUB
                                ;18307 ;*************************************************************************
                                ;18308
                                ;18309 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;18310 =00000
                                ;18311 =01111
                                ;18312 VF.CMPZV:
                                ;18313         ;01111-------------------------; ED
                                ;18314         PUSH,SET FLAG1,SIZE[IDEP],     ;
                                ;18315         SL_D_SEXT(M[MDR]) WBRANGE?,    ; SL <- D <- SIZE & FLAG<2:0> <- 2
U 024F, 0849,2B56,6DFD,8047,0D7E,8 324*  ;18316         NEXT7VF.VSUB                   ; BUT ON RANGE OF SIZE TO VIELD SUB
                                ;18317
                                ;18318 VF.CMPZV.1:
                                ;18319         ;10000-------------------------;
                                ;18320         WB_R[TEMP2]~M[MDR],            ;
U 0250, 0481,2003,0130,8847,003F,9       ;18321         SIZE[IDEP],CCOP1,IRD1          ; SET CC ON FIELD - SRC
                                ;18322 =
                                ;18323
                                ;18324
                                ;18325 =00000
                                ;18326 =01111
                                ;18327 VF.CMPV:
                                ;18328         ;01111-------------------------; EC
                                ;18329         PUSH,SET FLAG2,SIZE[IDEP],     ;
                                ;18330         SL_D_SEXT(M[MDR]) WBRANGE?,    ; SL <- D <- SIZE & FLAG<2:0> <- 4
U 026F, 0851,2B56,6DFD,8047,0D7E,8 324*  ;18331         NEXT7VF.VSUB                   ; BUT ON RANGE OF SIZE TO VIELD SUB
                                ;18332
                                ;18333         ;10000-------------------------;
                                ;18334         M[TEMP2]_MB.ASR.P,             ; SIGN EXTEND FIELD
U 0270, 0886,20F7,0030,0047,0025,0       ;18335         NEXT/VF.CMPZV.1                ; & GO DO COMPARE
                                ;18336 =
```

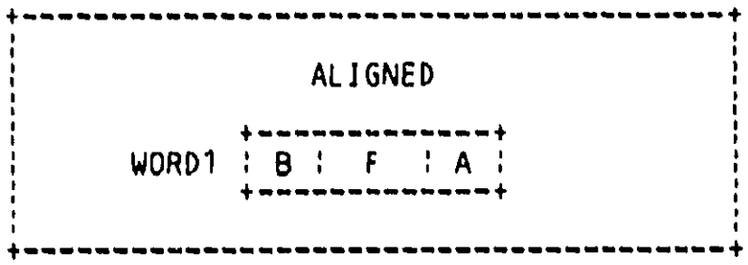
```
                                ;18337 .TOC   ""      Variable Length Bit Field      : VF.VSUB""
                                ;18338
                                ;18339 ;*****************************************************************************
                                ;18340 ;       Input           SLATCH,D        Size
                                ;18341 ;                       RNUM            Register number of Base
                                ;18342 ;                       OSR             Base Os
                                ;18343 ;                       Q               Startpos/Pos - Junk on Exit
                                ;18344 ;                       FLAG<2:0>=000   EXTV
                                ;18345 ;                       FLAG<2:0>=100   CMPV
                                ;18346 ;                       FLAG<2:0>=001   FFS, FFC, EXTZV
                                ;18347 ;                       FLAG<2:0>=010   CMPZV
                                ;18348 ;
                                ;18349 ;       Output          TEMP1           Position
                                ;18350 ;                       TEMP2           Extracted Field
                                ;18351 ;                       FLAG3           Size = 32
                                ;18352 ;                       PLATCH          -Size for EXTV and CMPV
                                ;18353 ;
                                ;18354 ;       Resources       PLATCH          Position
                                ;18355 ;                       MDR             0
                                ;18356 ;                       TEMP4           Work
                                ;18357 ;                       VA              Address of ms longword of field
                                ;18358 ;*****************************************************************************
                                ;18359 .REGION/VIELD.R1L,VIELD.R1H/VIELD.R2L,VIELD.R2H/VIELD.R3L,VIELD.R3H
                                ;18360 =000
                                ;18361 VF.VSUB:
                                ;18362         ;000-----------------------------; SIZE.EQ.[1,31]
                                ;18363         M[TEMP1]_PL_Q,MDR_0,              ; TEMP1 <- POS, MDR <- 0
U 17E8, 0486,1BFA,493D,84E7,0022,8  ;18364     REG MODE?,NEXT/VF.VSUB.MEM        ; REGISTER MODE?
                                ;18365
                                ;18366         ;001-----------------------------; SIZE.EQ.0
                                ;18367         PUSH,M[TEMP1]_PL_Q,               ; TEMP1 <- POS, MDR <- 0
U 17E9, 0486,1BFA,463D,8047,0413,0  ;18368     BRA ON ADD?,NEXT/OS.VADD          ;
                                ;18369
                                ;18370         ;010-----------------------------; SIZE.EQ.32
                                ;18371         M[TEMP1]_PL_Q,MDR_0,SET FLAG3,    ; TEMP1 <- POS, MDR <- 0
U 17EA, 045E,1BFA,493D,84E7,0022,8  ;18372     REG MODE?,NEXT/VF.VSUB.MEM        ; SET SIZE.EQ.32 FLAG
                                ;18373
                                ;18374         ;011-----------------------------; SIZE.GT.32
U 17EB, 0C80,0036,4030,0047,00FF,8  ;18375     NOP,NEXT/IE.OPER.FAULT            ; RESERVED OPERAND
                                ;18376
                                ;18377         ;100-----------------------------; RET+3 : SIZE.EQ.0
                                ;18378         M[TEMP2]_R[ZERO],                 ; EXTRACTED FIELD = 0
U 17EC, 0C86,25BE,467D,8047,017E,5  ;18379     IR<2-0>?,NEXT/VF.VSUB.SIZ0        ; (FF,EXT) OR CMP?
                                ;18380
                                ;18381         ;101-----------------------------; RET+4 : SIZE.EQ.0
                                ;18382         M[TEMP2]_R[ZERO],                 ; EXTRACTED FIELD = 0
U 17ED, 0C86,25BE,467D,8047,017E,5  ;18383     IR<2-0>?                          ; (FF,EXT) OR CMP?
                                ;18384 =
                                ;18385 =101
                                ;18386 VF.VSUB.SIZ0:
                                ;18387         ;101-----------------------------; CMP
U 17E5, 0C80,0036,41B0,0047,0010,0  ;18388     LOD INC BRA?,NEXT/OS.RED          ; OS.RED WILL DO RETURN FROM VSUB
                                ;18389
                                ;18390         ;111-----------------------------; FF OR EXT
U 17E7, 0480,0036,41B0,0047,0014,0  ;18391     LOD INC BRA?,NEXT/OS.WRT1         ; OS.WRT1 WILL DO RETURN FROM VSUB
```

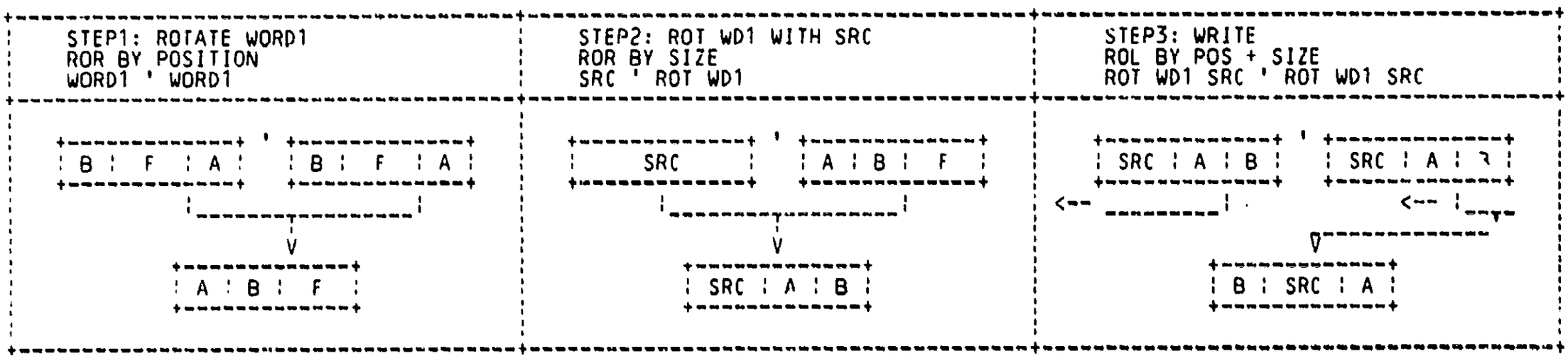
```
                                ;18392 =00
                                ;18393 VF.VSUB.MEM:
                                ;18394          ;00-------------------------------; MEM : SIZE.EQ.[1,32]
                                ;18395          PUSH,R[TEMP4]_M[TEMP1].ASR.3,   ; SET UP FOR ALLIGNED MEM REF
U 0228, 0084,1777,0631,0047,0413,0  ;18396      BRA ON ADD?,NEXT/OS.VADD        ; RET IS ABOVE FROM PREVIOUS PUSH
                                ;18397
                                ;18398 VF.VSUB.REG:
                                ;18399          ;01-------------------------------; REG : SIZE.EQ.[1,32]
                                ;18400          WB_R[TEMP1].SR.5(IF MDR IS  0), ; (NOTE : MDR WAS SET TO 0 ABOVE)
                                ;18401          ALUS_SIGND,SIZE[LONG],          ; ALUS <- (POS < 32)
U 0229, 0481,2251,8520,5807,0024,4  ;18402      FLAG<1-0>?,NEXT/VF.VSUB.REG1    ;
                                ;18403 =11
                                ;18404          ;11-------------------------------; MEM : RET+3 : SIZE.EQ.[1,31]
                                ;18405          VA_M[MDR]+R[TEMP4],PL<4-3>_WB,  ; VA <- LONGWORD ADD OF FIELD
U 022B, 0081,2FC1,0531,04A7,0023,4  ;18406      FLAG<1-0>?                      ; PL <- POS IN LONGWRD OF STR OF FLD
                                ;18407 =00
                                ;18408          ;00-------------------------------; EXTV,CMPV
                                ;18409          READ.LONG,VA_VA+4,              ; READ 1ST LONGWORD
U 0234, 0080,0036,4DF0,0451,017E,2  ;18410       (PL+SL).GT.32?,NEXT/VF.VSUB.MEM2; ALLIGNED?
                                ;18411
                                ;18412          ;01-------------------------------; FFS,FFC,EXTZV
                                ;18413          READ.LONG,VA_VA+4,              ; READ 1ST LONGWORD
U 0235, 0080,0036,4DF0,0451,017E,E  ;18414       (PL+SL).GT.32?,NEXT/VF.VSUB.MEM3; ALLIGNED?
                                ;18415
                                ;18416          ;10-------------------------------; CMPZV
                                ;18417          READ.LONG,VA_VA+4,              ; READ 1ST LONGWORD
U 0236, 0080,0036,4DF0,0451,0003,6  ;18418       (PL+SL).GT.32?,NEXT/VF.VSUB.MEM4; ALLIGNED?
                                ;18419 =
                                ;18420 =10
                                ;18421 VF.VSUB.MEM2:
                                ;18422          ;10-------------------------------; EXTV,CMPV : ALLIGNED
                                ;18423          R[TEMP2]_M[MDR].RR.PS,          ; TEMP2 <- FIELD
U 17E2, 0085,2977,04B0,8047,0000,3  ;18424       FLAG2?,NEXT/VF.VSUB.REG5        ; CMPV OR EXTV ?
                                ;18425
                                ;18426          ;11-------------------------------; EXTV,CMPV : UNALLIGNED
U 17E3, 0085,2592,4030,8051,0038,3  ;18427       R[TEMP2]_M[MDR],READ.LONG       ; SAVE 1ST LONGWORD & READ 2ND
                                ;18428
                                ;18429          ;-------------------------------; EXTV,CMPV : UNALLIGNED
                                ;18430          R[TEMP2]_(M[MDR]_RB).RR.PS,     ; TEMP2 <- FIELD
U 0383, 0485,2937,04B0,8047,0000,3  ;18431       FLAG2?,NEXT/VF.VSUB.REG5        ; CMPV OR EXTV ?
                                ;18432 =10
                                ;18433 VF.VSUB.MEM3:
                                ;18434          ;10-------------------------------; FFS,FFC,EXTZV : ALLIGNED
                                ;18435          R[TEMP2]_M[MDR].XZ,              ; TEMP2 <- FIELD
U 17EE, 0085,2077,01B0,8047,0014,0  ;18436       LOD INC BRA?,NEXT/OS.WRT1       ; OS.WRT1 WILL DO RET FROM VSUB
                                ;18437
                                ;18438          ;11-------------------------------; FFS,FFC,EXTZV : UNALLIGNED
U 17EF, 0885,2592,4030,8051,0038,4  ;18439       R[TEMP2]_M[MDR],READ.LONG       ; SAVE 1ST LONGWORD & READ 2ND
                                ;18440
                                ;18441          ;-------------------------------;
                                ;18442          R[TEMP2]_(M[MDR]_RB).XZ,        ; TEMP2 <- FIELD
U 0384, 0485,2037,01B0,8047,0014,0  ;18443       LOD INC BRA?,NEXT/OS.WRT1       ; OS.WRT1 WILL DO RET FROM VSUB
```

```
                                ;18444 =10
                                ;18445 VF.VSUB.MEM4:
                                ;18446          ;10-------------------------------; CMPZV : ALLIGNED
                                ;18447          R[TEMP2]_M[MDR].XZ,               ; TEMP2 <- FIELD
U 0036, 0885,2077,01B0,8047,0010,0  ;18448      LOD INC BRA?,NEXT/OS.RED          ; OS.RED WILL DO RET FROM VSUB
                                ;18449
                                ;18450          ;11-------------------------------; CMPZV : UNALLIGNED
U 0037, 0085,2592,4030,8051,0038,5  ;18451      R[TEMP2]_M[MDR],READ.LONG         ; SAVE 1ST LONGWORD & READ 2ND
                                ;18452
                                ;18453          ;-------------------------------;
                                ;18454          R[TEMP2] (M[MDR] RB).XZ,          ; TEMP2 <- FIELD
U 0385, 0C85,2037,01B0,8047,0010,0  ;18455      LOD INC BRA?,NEXT/OS.RED          ; OS.RED WILL DO RET FROM VSUB
                                ;18456 =00
                                ;18457 VF.VSUB.REG1:
                                ;18458          ;00-------------------------------; EXTV,CMPV
                                ;18459          M[TEMP2] R[GPR.R+1] Q_MB,         ; SET UP IN CASE UNALLIGNED
                                ;18460          POS>31? (PL+SL)>32?,              ; POS>31? OR UNALLIGNED?
U 0244, 0086,2184,7DBF,D847,0024,8  ;18461      NEXT/VF.VSUB.REG2                 ;
                                ;18462
                                ;18463          ;01-------------------------------; FFS,FFC,EXTZV
                                ;18464          M[TEMP2] R[GPR.R+1] Q_MB,         ; SET UP IN CASE UNALLIGNED
                                ;18465          POS>31? (PL+SL)>32?,              ; POS>31? OR UNALLIGNED?
U 0245, 0486,2184,7DBF,D847,0025,4  ;18466      NEXT/VF.VSUB.REG3                 ;
                                ;18467
                                ;18468          ;10-------------------------------; CMPZV
                                ;18469          M[TEMP2] R[GPR.R+1] Q_MB,         ; SET UP IN CASE UNALLIGNED
                                ;18470          POS>31? (PL+SL)>32?,              ; POS>31? OR UNALLIGNED?
U 0246, 0C86,2184,7DBF,D847,0025,C  ;18471      NEXT/VF.VSUB.REG4                 ;
                                ;18472 =
                                ;18473 =00
                                ;18474 VF.VSUB.REG2:
                                ;18475          ;00-------------------------------; EXTV,CMPV : POS > 31
U 0248, 0C80,0036,4030,0047,00FF,8  ;18476      NOP,NEXT/IE.OPER.FAULT            ;
                                ;18477
                                ;18478          ;01-------------------------------; EXTV,CMPV : POS > 31
U 0249, 0C80,0036,4030,0047,00FF,8  ;18479      NOP,NEXT/IE.OPER.FAULT            ;
                                ;18480
                                ;18481          ;10-------------------------------; EXTV,CMPV : ALLIGNED
                                ;18482          M[TEMP2] R[GPR.R].RR.PS,          ; TEMP2 <- FIELD
U 024A, 0486,29B7,04BC,C047,0000,3  ;18483      FLAG2?,NEXT/VF.VSUB.REG5          ; EXTV OR CMPV?
                                ;18484
                                ;18485          ;11-------------------------------; EXTV,CMPV : UNALLIGNED
                                ;18486          M[TEMP2] (MB R[GPR.R]).RR.PS,     ; TEMP2 <- FIELD
U 024B, 0086,2937,04BC,C047,0000,3  ;18487      FLAG2?,NEXT/VF.VSUB.REG5          ; EXTV OR CMPV?
```

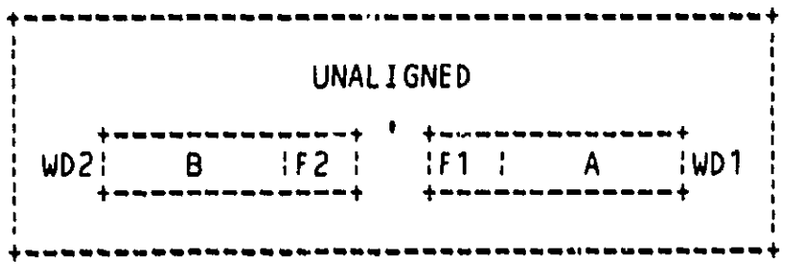
```
                                ;18488 =00
                                ;18489 VF.VSUB.REG3:
                                ;18490          ;00-------------------------------; FFS,FFC,EXTZV : POS > 31
U 0254, 0C80,0036,4030,0047,00FF,8 ;18491       NOP,NEXT/IE.OPER.FAULT          ;
                                ;18492
                                ;18493          ;01-------------------------------; FFS,FFC,EXTZV : POS > 31
U 0255, 0C80,0036,4030,0047,00FF,8 ;18494       NOP,NEXT/IE.OPER.FAULT          ;
                                ;18495
                                ;18496          ;10-------------------------------; FFS,FFC,EXTZV : ALLIGNED
                                ;18497          M[TEMP2] R[GPR.R].XZ,           ; TEMP2 <- FIELD
U 0256, 0486,20B7,01BC,C047,0014,0 ;18498       LOD INC BRA?,NEXT/OS.WRT1       ; OS.WRT1 WILL DO RET FROM VSUB
                                ;18499
                                ;18500          ;11-------------------------------; FFS,FFC,EXTZV : UNALLIGNED
                                ;18501          M[TEMP2] (MB R[GPR.R]).XZ,      ; TEMP2 <- FIELD
U 0257, 0086,2037,01BC,C047,0014,0 ;18502       LOD INC BRA?,NEXT/OS.WRT1       ; OS.WRT1 WILL DO RET FROM VSUB
                                ;18503 =00
                                ;18504 VF.VSUB.REG4:
                                ;18505          ;00-------------------------------; CMPZV : POS > 31
U 025C, 0C80,0036,4030,0047,00FF,8 ;18506       NOP,NEXT/IE.OPER.FAULT          ;
                                ;18507
                                ;18508          ;01-------------------------------; CMPZV : POS > 31
U 025D, 0C80,0036,4030,0047,00FF,8 ;18509       NOP,NEXT/IE.OPER.FAULT          ;
                                ;18510
                                ;18511          ;10-------------------------------; CMPZV : ALLIGNED
                                ;18512          M[TEMP2] R[GPR.R].XZ,           ; TEMP2 <- FIELD
U 025E, 0C86,20B7,01BC,C047,0010,0 ;18513       LOD INC BRA?,NEXT/OS.RED        ; OS.RED WILL DO RET FROM VSUB
                                ;18514
                                ;18515          ;11-------------------------------; CMPZV : UNALLIGNED
                                ;18516          M[TEMP2] (MB R[GPR.R]).XZ,      ; TEMP2 <- FIELD
U 025F, 0886,2037,01BC,C047,0010,0 ;18517       LOD INC BRA?,NEXT/OS.RED        ; OS.RED WILL DO RET FROM VSUB
                                ;18518 =0**
                                ;18519 VF.VSUB.REG5:
                                ;18520          ;0**------------------------------; EXTV
U 0003, 0080,0BE3,01BD,8047,0014,0 ;18521       PL_-D,LOD INC BRA?,NEXT/OS.WRT1 ; PL <- -SIZE. OS.WRT1 WILL RET FRM VSUB
                                ;18522
                                ;18523          ;1**------------------------------; CMPV
U 0007, 0880,0BE3,01BD,8047,0010,0 ;18524       PL_-D,LOD INC BRA?,NEXT/OS.RED  ; PL <- -SIZE. OS.RED WILL RET FRM VSUB
```

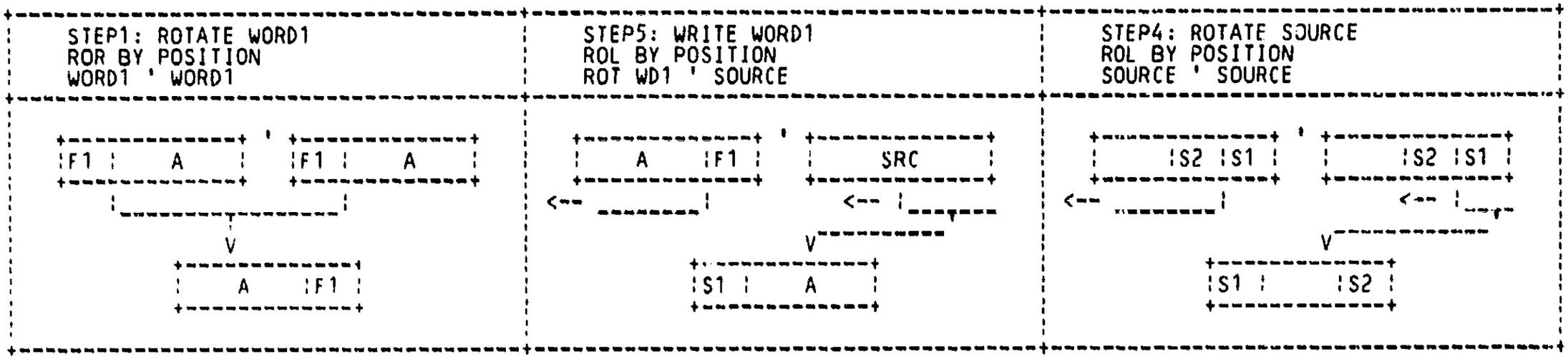
```
                                ;18525 .TOC     ""      Variable Length Bit Field       : Algorithm For INSV Alligned""
;18526  .NOBIN
;18527
;18528
;18529
;18530
;18531
;18532
;18533
;18534  ;
;18535  ;
;18536  ;
;18537  ;
;18538  ;
;18539  ;
;18540  ;
;18541  ;
;18542  ;
;18543  ;
;18544  ;
;18545  ;
;18546  ;
;18547  ;
;18548  ;
;18549  ;
;18550  ;
;18551  ;
;18552  ;
;18553  ;
;18554  ;
;18555  ;
;18556  ;
;18557  ;
;18558  ;
;18559  ;
;18560  ;
;18561  ;
;18562  ;
;18563  ;
;18564  ;
```

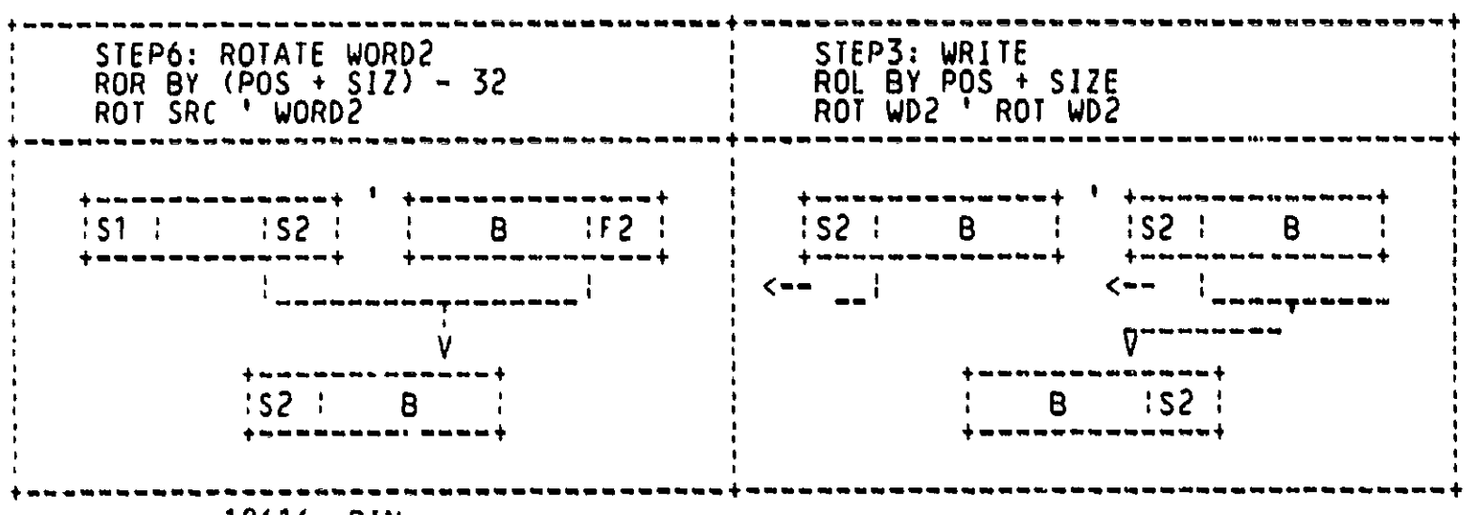




```
;18565  .TOC     ""      Variable Length Bit Field       : Algorithm For INSV Un-Alligned""
;18566
;18567  ;
;18568  ;
;18569  ;
;18570  ;
;18571  ;
;18572  ;
;18573  ;
;18574  ;
;18575  ;
;18576  ;
;18577  ;
;18578  ;
;18579  ;
;18580  ;
;18581  ;
;18582  ;
;18583  ;
;18584  ;
;18585  ;
;18586  ;
;18587  ;
;18588  ;
;18589  ;
;18590  ;
;18591  ;
;18592  ;
;18593  ;
;18594  ;
;18595  ;
;18596  ;
;18597  ;
;18598  ;
;18599  ;
;18600  ;
;18601  ;
;18602  ;
;18603  ;
;18604  ;
;18605  ;
;18606  ;
;18607  ;
;18608  ;
;18609  ;
;18610  ;
;18611  ;
;18612  ;
;18613  ;
;18614  ;
;18615  ;
```








```
;18616 .BIN
```

```
                                ;18617 .TOC   "      Variable Length Bit Field       : INSV"
                                ;18618
                                ;18619 ;*****************************************************************************
                                ;18620 ;       INSV   src.rl pos.rl size.rb base.ab
                                ;18621 ;
                                ;18622 ;          Input         MDR              Position
                                ;18623 ;                        Q                Source
                                ;18624 ;
                                ;18625 ;          Resources     SLATCH  Size
                                ;18626 ;                        PLATCH           Position
                                ;18627 ;                        ALUS<0>          Set to 1 by OS.VADD/Reg Mode
                                ;18628 ;                                         when POS(TEMP1) > 31
                                ;18629 ;
                                ;18630 ;                        +----------------------+----------------------+
                                ;18631 ;                        |       MEMORY         |       REGISTER       |
                                ;18632 ;                        +----------+-----------+----------+-----------+
                                ;18633 ;                        | ALLIGNED | UNALLIGNED| ALLIGNED | UNALLIGNED|
                                ;18634 ;                        +----------+-----------+----------+-----------+
                                ;18635 ;              TEMP0     |          | Work      |          |           |
                                ;18636 ;              TEMP1     | Position | Position  | Position | Position  |
                                ;18637 ;              TEMP2     |          | Work      |          |           |
                                ;18638 ;              TEMP3     | Work     | Work      |          |           |
                                ;18639 ;              TEMP5,D   | Source   | Source    | Source   | Source    |
                                ;18640 ;              TEMP6     | Work     | Work      |          |           |
                                ;18641 ;              FLAG0     |          |           | 0        | 1         |
                                ;18642 ;              FLAG3     | SIZE=32  | SIZE=32   | SIZE=32  | SIZE=32   |
                                ;18643 ;                        +----------+-----------+----------+-----------+
                                ;18644 ;
                                ;18645 ;*****************************************************************************
                                ;18646
                                ;18647 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;18648 =000
                                ;18649 VF.INSV:
                                ;18650          ;000-------------------------------; F0
                                ;18651          PUSH,M[TEMP5]_D_Q,                ; TEMP5 & D <- SOURCE
U 0280, 0086,503A,61BD,8047,0410,0 ;18652       LOD INC BRA?,NEXT/OS.RED          ; GO EVALUATE SIZE
                                ;18653
                                ;18654          ;001-------------------------------; RET+1
                                ;18655          PUSH,                             ;
U 0281, 0C81,2B42,4DFD,8047,0429,8 ;18656       SL_M[MDR] WBRANGE?,NEXT/VF.INSV1; SL <- SIZE
                                ;18657 =100
                                ;18658          ;100-------------------------------; RET+3 : MEMORY
                                ;18659          R[TEMP6]_M[TEMP1].ASR.3,          ; TEMP6 <- POS<31:3>(LONGWORD OFFSET
U 0284, 0C84,1777,0031,8047,0038,6 ;18660       NEXT/VF.INSV.MEM                  ;                    OF POSITION)
                                ;18661
                                ;18662          ;101-------------------------------; RET+4 : REGISTER
                                ;18663          R[GPR.R]_RB.RR.P,                 ; STEP1 : ROTATE WORD 1
                                ;18664          POS>31? (PL+SL)>32?,              ;
U 0285, 0084,01B7,0DBC,D847,0026,8 ;18665       NEXT/VF.INSV.REG                  ;
                                ;18666 =
```

```
                                    ;18667 .REGION/VIELD.R1L,VIELD.R1H/VIELD.R2L,VIELD.R2H/VIELD.R3L,VIELD.R3H
                                    ;18668 =000
                                    ;18669 VF.INSV1:
                                    ;18670          ;000--------------------------------; SIZE.EQ.[1,31]
                                    ;18671          M[TEMP1]_PL_Q,MDR_0,                ; T1 <- POS, MDR_0 FOR OS.VADD REG MODE
U 0298, 0486,1BFA,41BD,84E7,0013,0  ;18672          LOD INC BRA?,NEXT/OS.VADD           ; GO EVALUATE BASE
                                    ;18673
                                    ;18674          ;001--------------------------------; SIZE.EQ.0
U 0299, 0480,0036,41B0,0047,0413,0  ;18675          PUSH,LOD INC BRA?,NEXT/OS.VADD      ; GO EVALUATE BASE
                                    ;18676
                                    ;18677          ;010--------------------------------; SIZE.EQ.32
                                    ;18678          M[TEMP1]_PL_Q,MDR_0,SET FLAG3,      ; T1 <- POS, MDR_0 FOR OS.VADD REG MODE
U 029A, 045E,1BFA,41BD,84E7,0013,0  ;18679          LOD INC BRA?,NEXT/OS.VADD           ; SET SIZE=32 FLAG, GO EVALUATE BASE
                                    ;18680
                                    ;18681          ;011--------------------------------; SIZE.GT.32
U 029B, 0C80,0036,4030,0047,00FF,8  ;18682          NOP,NEXT/IE.OPER.FAULT              ; RESERVED OPERAND
                                    ;18683
                                    ;18684          ;100--------------------------------; RET+3 : SIZE.EQ.0
U 029C, 0480,0E7C,013D,8047,003F,9  ;18685          WB_R[ZERO]+1,IRD1                   ; SET CC'S AND DONE
                                    ;18686
                                    ;18687          ;101--------------------------------; RET+4 : SIZE.EQ.0
U 029D, 0480,0E7C,013D,8047,003F,9  ;18688          WB_R[ZERO]+1,IRD1                   ; SET CC'S AND DONE
                                    ;18689 =
                                    ;18690 VF.INSV.MEM:
                                    ;18691          ;--------------------------------; SIZE.NE.0
U 0386, 0085,2FC1,0031,84A7,0038,7  ;18692          VA_R[TEMP6]_M[MDR] RB,PL<4-3>_WB; VA<-LONG ADD OF FLD, PL<-POS OF FLD
                                    ;18693
                                    ;18694          ;--------------------------------;
U 0387, 0080,0036,4D:0,0055,0004,2  ;18695          READ.LONG.MOD,(PL+SL).GT.32?        ; READ 1ST LONGWORD, ALLIGNED?
                                    ;18696 =10
                                    ;18697          ;10---------------------------------; ALLIGNED
                                    ;18698          R[TEMP3]_M[MDR].RR.P,               ; STEP1 : ROTATE WORD 1
U 0042, 0C85,2177,04F0,C047,0027,2  ;18699          FLAG3?,NEXT/VF.INSV.MEM.ALIND       ; SIZE=32?
                                    ;18700
                                    ;18701          ;11---------------------------------; UNALLIGNED
                                    ;18702          R[TEMP0]_M[TEMP5].RL.P,VA_VA+4,     ; STEP4 : ROTATE SOURCE
U 0043, 0084,5877,0030,0447,0038,9  ;18703          NEXT/VF.INSV.MEM.UNALIND            ; BUMP VA FOR 2ND READ
                                    ;18704 =0
                                    ;18705 VF.INSV.MEM.ALIND:
                                    ;18706          ;0---------------- ----------; ALLIGNED, SIZE.NE.32
                                    ;18707          R[TEMP3]_(M[TEMP5] RB).RR.S,        ; STEP2 : ROTATE WORD 1 WITH SOURCE
U 0272, 0484,51F7,0030,C047,0038,E  ;18708          NEXT/VF.INSV.MEM1                   ;
                                    ;18709
                                    ;18710          ;1----------------------------------; ALLIGNED,SIZE[IDEP], SIZE.EQ.32
U 0273, 0080,0022,413D,8559,003F,9  ;18711          WRITE.LONG D,SIZE[IDEP],IRD1        ; WRITE SOURCE
```

```
                                ;18712 VF.INSV.MEM.UNALIND:
                                ;18713          ;-------------------------------; UNALLIGNED
                                ;18714          R[TEMP2]_M[MDR].RR.P,           ; STEP1 : ROTATE WORD 1
U 0389, 0C85,2177,0030,8055,0038,A ;18715       READ.LONG.MOD                   ; READ 2ND LONGWORD
                                ;18716
                                ;18717          ;-------------------------------; UNALLIGNED
U 038A, 0080,6002,403D,84A7 0038,B ;18718       VA_M[TEMP6]                     ; SET VA TO 1ST LONGWORD FOR WRITE
                                ;18719
                                ;18720          ;-------------------------------; UNALLIGNED
                                ;18721          SET MM.NOINT,SIZE[IDEP],        ;
                                ;18722          WRITE.LONG,                     ; STEP5 : WRITE 1ST WORD
U 038B, 0460,58F7,0030,8559,0038,C ;18723       WB_(R[TEMP2] M[TEMP5]).RL.P     ;
                                ;18724
                                ;18725          ;-------------------------------; UNALLIGNED
U 038C, 0C85,2592,4030,C447,0038,D ;18726       R[TEMP3]_M[MDR],VA_VA+4         ;
                                ;18727
                                ;18728          ;-------------------------------; UNALLIGNED
U 038D, 0084,0937,0030,C047,0038,E ;18729       R[TEMP3]_(M[TEMP0] RB).RR.PS    ; STEP6 : ROTATE WORD 2
                                ;18730
                                ;18731 VF.INSV.MEM1:
                                ;18732          ;-------------------------------;
                                ;18733          WRITE.LONG,SIZE[IDEP],IRD1,     ; STEP3 : WRITE
U 038E, 0480,3837,0130,C559,003F,9 ;18734       WB_(R[TEMP3] M[TEMP3]).RL.PS    ;
```

```
                                ;18735 =00
                                ;18736 VF.INSV.REG:
                                ;18737          ;00-------------------------------; POS.GT.31, ALLIGNED
                                ;18738          R[GPR.R]_RB.RL.P,                 ; UN-DO STEP1 THE TURKEY BLEW IT
U 0268, 0884,08B7,003C,C047,00FF,8 ;18739       NEXT/IE.OPER.FAULT                ;
                                ;18740
                                ;18741          ;01-------------------------------; POS.GT.31, UNALLIGNED
                                ;18742          R[GPR.R]_RB.RL.P,                 ; UN-DO STEP1 THE TURKEY BLEW IT
U 0269, 0884,08B7,003C,C047,00FF,8 ;18743       NEXT/IE.OPER.FAULT                ;
                                ;18744
                                ;18745          ;10-------------------------------; POS.LE.31, ALL ONE
                                ;18746          M[TEMP5]_(MB R[GPR.R]).RR.S,      ; STEP2 : ROTATE WITH SOURCE
U 026A, 0486,51F7,04FC,C047,0027,A ;18747       FLAG3?,NEXT/VF.INSV.REG1          ; SIZE=32?
                                ;18748
                                ;18749          ;11-------------------------------; POS.LE.31, UNALLIGNED
                                ;18750          R[GPR.R]_(RB M[TEMP5]).RL.P,      ; STEP5 : WRITE WORD 1
U 026B, 0C44,58F7,C03C,C047,0038,7 ;18751       SET FLAG0                         ; SET UNALLIGNED FLAG
                                ;18752
                                ;18753          ;-------------------------------; UNALLIGNED
U 038F, 0084,5877,0031,4047,0039,5 ;18754       R[TEMP5]_M[TEMP5].RL.P            ; STEP4 : ROTATE SOURCE
                                ;18755
                                ;18756          ;-------------------------------; UNALLIGNED
U 0395, 0486,5937,003F,C047,0027,A ;18757       M[TEMP5]_(MB R[GPR.R+1]).RR.PS    ; STEP6 : ROTATE WORD 2
                                ;18758 =0
                                ;18759 VF.INSV.REG1:
                                ;18760          ;0-------------------------------; SIZE.NE.32(CHECKED FOR ALLIGNED ONLY)
                                ;18761          M[TEMP5]_(R[TEMP5] MB).RL.PS,     ; STEP3 : WRITE
U 027A, 0C86,5837,0431,4047,0028,2 ;18762       FLAG0?,NEXT/VF.INSV.REG2          ; UNALLIGNED?
                                ;18763
                                ;18764          ;1-------------------------------; SIZE.EQ.32, ALLIGNED
U 027B, 0484,05B2,413C,C047,003F,9 ;18765       R[GPR.R]_D,IRD1                   ; WRITE SOURCE
                                ;18766 =0
                                ;18767 VF.INSV.REG2:
                                ;18768          ;0-------------------------------; ALLIGNED
U 0282, 0084,5592,413C,C047,003F,9 ;18769       R[GPR.R]_M[TEMP5],IRD1            ; WRITE
                                ;18770
                                ;18771          ;1-------------------------------; UNALLIGNED
U 0283, 0084,5592,413F,C047,003F,9 ;18772       R[GPR.R+1]_M[TEMP5],IRD1          ; WRITE 2ND WORD
```

```
                                        ;18773 .TOC     "      Variable Length Bit Field       : Ird Rom Definition"
;18774  .NOBIN
;18775
;18776  .ICODE  ;       OPS             REG                     MEM             OPS         FPA REG                 FPA MEM
;18777       OEC: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                 ;CMPV
;18778            IRD1[LOD][OS.RED              ]                                [LOD][OS.RED            ]
;18779  .OCODE
;18780       OEC: CNT0[LOD][OS.RED              ][OS.RED            ]  [LOD][OS.RED            ][OS.RED            ],
;18781            CNT1[LOD][VF.CMPV             ][VF.CMPV           ]  [LOD][VF.CMPV           ][VF.CMPV           ]
;18782
;18783  .ICODE
;18784       OED: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                 ;CMPZV
;18785            IRD1[LOD][OS.RED              ]                                [LOD][OS.RED            ]
;18786  .OCODE
;18787       OED: CNT0[LOD][OS.RED              ][OS.RED            ]  [LOD][OS.RED            ][OS.RED            ],
;18788            CNT1[LOD][VF.CMPZV            ][VF.CMPZV          ]  [LOD][VF.CMPZV          ][VF.CMPZV          ]
;18789
;18790  .ICODE
;18791       OEE: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                 ;EXTV
;18792            IRD1[LOD][OS.RED              ]                                [LOD][OS.RED            ]
;18793  .OCODE
;18794       OEE: CNT0[LOD][OS.RED              ][OS.RED            ]  [LOD][OS.RED            ][OS.RED            ],
;18795            CNT1[LOD][VF.EXTV             ][VF.EXTV           ]  [LOD][VF.EXTV           ][VF.EXTV           ]
;18796
;18797  .ICODE
;18798       OEF: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                 ;EXTZV
;18799            IRD1[LOD][OS.RED              ]                                [LOD][OS.RED            ]
;18800  .OCODE
;18801       OEF: CNT0[LOD][OS.RED              ][OS.RED            ]  [LOD][OS.RED            ][OS.RED            ],
;18802            CNT1[LOD][VF.EXTZV            ][VF.EXTZV          ]  [LOD][VF.EXTZV          ][VF.EXTZV          ]
;18803
;18804  .ICODE
;18805       OEB: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                 ;FFC
;18806            IRD1[LOD][OS.RED              ]                                [LOD][OS.RED            ]
;18807  .OCODE
;18808       OEB: CNT0[LOD][OS.RED              ][OS.RED            ]  [LOD][OS.RED            ][OS.RED            ],
;18809            CNT1[LOD][VF.FFC              ][VF.FFC            ]  [LOD][VF.FFC            ][VF.FFC            ]
;18810
;18811  .ICODE
;18812       OEA: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                 ;FFS
;18813            IRD1[LOD][OS.RED              ]                                [LOD][OS.RED            ]
;18814  .OCODE
;18815       OEA: CNT0[LOD][OS.RED              ][OS.RED            ]  [LOD][OS.RED            ][OS.RED            ],
;18816            CNT1[LOD][VF.FFS              ][VF.FFS            ]  [LOD][VF.FFS            ][VF.FFS            ]
;18817
;18818  .ICODE
;18819       OF0: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                 ;INSV
;18820            IRD1[LOD][OS.RED              ]                                [LOD][OS.RED            ]
;18821  .OCODE
;18822       OF0: CNT0[LOD][OS.RED              ][OS.RED            ]  [LOD][OS.RED            ][OS.RED            ],
;18823            CNT1[NOP][VF.INSV             ][VF.INSV           ]  [NOP][VF.INSV           ][VF.INSV           ]
```

```
;18824  .TOC   ''      Variable Length Bit Field      : Dsize Rom Definition''
;18825  .DCODE
;18826
;18827  OEC:    SIZE [LONG] [BYTE] [BYTE] [LONG] [   0] [   0]  ;CMPV
;18828
;18829  OED:    SIZE [LONG] [BYTE] [BYTE] [LONG] [   0] [   0]  ;CMPZV
;18830
;18831  OEE:    SIZE [LONG] [BYTF] [BYTE] [LONG] [   0] [   0]  ;EXTV
;18832
;18833  OEF:    SIZE [LONG] [BYTE] [BYTE] [LONG] [   0] [   0]  ;EXTZV
;18834
;18835  OEB:    SIZE [LONG] [BYTE] [BYTE] [LONG] [   0] [   0]  ;FFC
;18836
;18837  OEA:    SIZE [LONG] [BYTE] [BYTE] [LONG] [   0] [   0]  ;FFS
;18838
;18839  OFO:    SIZE [LONG] [LONG] [BYTE] [BYTE] [   0] [   0]  ;INSV
;18840
;18841  .UCODE
                                        ;18842 .BIN
```

18842; This page intentionally left blank.

```
                                        ;18843 .TOC "CONTRL.MIC"
                                        ;18844 .TOC "Revision 18.0"
                                        ;18845 ;       G. koeckhoven, Brian Allison
                                        ;18846
;18847  .NOBIN
;18848  .TOC "  Revision History"
;18849
;18850  ; REV   EXPLAINATION
;18851  ; 18    Force some addresses
;18852  ; 17    Initial release.
                                        ;18853          .BIN
```

```
                                ;18854 .TOC   "       Control Instructions          : Conditional Branch Instructions"
                                ;18855
                                ;18856 ;*****************************************************************************
                                ;18857 ;       Conditional Branch Instructions
                                ;18858 ;
                                ;18859 ;       BXXXX   displ.bb
                                ;18860 ;          Input                          OSR             Displacement
                                ;18861 ;*****************************************************************************
                                ;18862 .REGION/IRD1.R1L,IRD1.R1H
                                ;18863 =1000
                                ;18864 CO.BRCND:
                                ;18865         ;1000-------------------------;
                                ;18866         MDR_ZEXT(OSR) BRATST?,        ; GET DISPLACEMENT FROM OSR
                                ;18867                                       ; TEST FOR BRANCH
U 0058, 0C80,0036,4B70,05E7,0072,2 ;18868      NEXT/CO.BRCND-DECIDE          ; GO TO DECISION BLOCK
                                ;18869
                                ;18870         ;1001-------------------------;
                                ;18871         MDR_ZEXT(OSR) BRATST?,        ; GET DISPLACEMENT FROM OSR
                                ;18872                                       ; TEST FOR BRANCH
U 0059, 0C80,0036,4B70,05E7,0072,2 ;18873      NEXT/CO.BRCND-DECIDE          ; GO TO DECISION BLOCK
                                ;18874
                                ;18875         ;1010-------------------------;
                                ;18876         MDR_ZEXT(OSR) BRATST?,        ; GET DISPLACEMENT FROM OSR
                                ;18877                                       ; TEST FOR BRANCH
U 005A, 0C80,0036,4B70,05E7,0072,2 ;18878      NEXT/CO.BRCND-DECIDE          ; GO TO DECISION BLOCK
                                ;18879
                                ;18880         ;1011-------------------------;
                                ;18881         MDR_ZEXT(OSR) BRATST?,        ; GET DISPLACEMENT FROM OSR
                                ;18882                                       ; TEST FOR BRANCH
U 005B, 0C80,0036,4B70,05E7,0072,2 ;18883      NEXT/CO.BRCND-DECIDE          ; GO TO DECISION BLOCK
                                ;18884
                                ;18885         ;1100-------------------------;
                                ;18886         MDR_ZEXT(OSR) BRATST?,        ; GET DISPLACEMENT FROM OSR
                                ;18887                                       ; TEST FOR BRANCH
U 005C, 0C80,0036,4B70,05E7,0072,2 ;18888      NEXT/CO.BRCND-DECIDE          ; GO TO DECISION BLOCK
                                ;18889
                                ;18890         ;1101-------------------------;
                                ;18891         MDR_ZEXT(OSR) BRATST?,        ; GET DISPLACEMENT FROM OSR
                                ;18892                                       ; TEST FOR BRANCH
U 005D, 0C80,0036,4B70,05E7,0072,2 ;18893      NEXT/CO.BRCND-DECIDE          ; GO TO DECISION BLOCK
                                ;18894
                                ;18895         ;1110-------------------------;
                                ;18896         MDR_ZEXT(OSR) BRATST?,        ; GET DISPLACEMENT FROM OSR
                                ;18897                                       ; TEST FOR BRANCH
U 005E, 0C80,0036,4B70,05E7,0072,2 ;18898      NEXT/CO.BRCND-DECIDE          ; GO TO DECISION BLOCK
                                ;18899
                                ;18900         ;1111-------------------------;
                                ;18901         MDR_ZEXT(OSR) BRATST?,        ; GET DISPLACEMENT FROM OSR
                                ;18902                                       ; TEST FOR BRANCH
U 005F, 0C80,0036,4B70,05E7,0072,2 ;18903      NEXT/CO.BRCND-DECIDE          ; GO TO DECISION BLOCK
                                ;18904
                                ;18905 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
```

```
                                ;18906 =0
                                ;18907 CO.NOP:
                                ;18908 CO.BRCND-DECIDE:
                                ;18909          ;0------------------------------; NO BRANCH IF CONTROL COMES HERE
U 0722, 0080,0036,4130,0047,003F,9 ;18910       IRD1                             ; GO DO NEXT INSTRUCTION
                                ;18911
                                ;18912 CO.BRCND-BRANCH:
                                ;18913          ;1------------------------------; BRANCH IF CONTROL COMES HERE !!
                                ;18914          PC_PC+SEXT(M[MDR]),              ; CALCULATE NEW PC
U 0723, 0C81,2816,403D,8587,0072,2 ;18915       SIZE[IDEP],NEXT/CO.NOP           ; WASTE A CYCLE TO LET PC CATCH UP
```

```
                                ;18916 .TOC     "       Control Instructions            : BRB"
                                ;18917
                                ;18918 ;*************************************************************************************
                                ;18919 ;       BRB     displ.bb
                                ;18920 ;         Input                                  OSR             Displacement
                                ;18921 ;*************************************************************************************
                                ;18922 .REGION/IRD1.R1L,IRD1.R1H
                                ;18923 =1000
                                ;18924 CO.BRB:
                                ;18925         ;1000-----------------------------;
                                ;18926         MDR_ZEXT(OSR),                  ; GET BRANCH DISPLACEMENT
U 0078, 0480,0036,4030,05E7,0072,3 ;18927      NEXT/CO.BRCND-BRANCH            ; FINNISH AT GOOD LEG OF COND BRANCH
                                ;18928
                                ;18929         ;1001-----------------------------;
                                ;18930         MDR_ZEXT(OSR),                  ; GET BRANCH DISPLACEMENT
U 0079, 0480,0036,4030,05E7,0072,3 ;18931      NEXT/CO.BRCND-BRANCH            ; FINNISH AT GOOD LEG OF COND BRANCH
                                ;18932
                                ;18933         ;1010-----------------------------;
                                ;18934         MDR_ZEXT(OSR),                  ; GET BRANCH DISPLACEMENT
U 007A, 0480,0036,4030,05E7,0072,3 ;18935      NEXT/CO.BRCND-BRANCH            ; FINNISH AT GOOD LEG OF COND BRANCH
                                ;18936
                                ;18937         ;1011-----------------------------;
                                ;18938         MDR_ZEXT(OSR),                  ; GET BRANCH DISPLACEMENT
U 007B, 0480,0036,4030,05E7,0072,3 ;18939      NEXT/CO.BRCND-BRANCH            ; FINNISH AT GOOD LEG OF COND BRANCH
                                ;18940
                                ;18941         ;1100-----------------------------;
                                ;18942         MDR_ZEXT(OSR),                  ; GET BRANCH DISPLACEMENT
U 007C, 0480,0036,4030,05E7,0072,3 ;18943      NEXT/CO.BRCND-BRANCH            ; FINNISH AT GOOD LEG OF COND BRANCH
                                ;18944
                                ;18945         ;1101-----------------------------;
                                ;18946         MDR_ZEXT(OSR),                  ; GET BRANCH DISPLACEMENT
U 007D, 0480,0036,4030,05E7,0072,3 ;18947      NEXT/CO.BRCND-BRANCH            ; FINNISH AT GOOD LEG OF COND BRANCH
                                ;18948
                                ;18949         ;1110-----------------------------;
                                ;18950         MDR_ZEXT(OSR),                  ; GET BRANCH DISPLACEMENT
U 007E, 0480,0036,4030,05E7,0072,3 ;18951      NEXT/CO.BRCND-BRANCH            ; FINNISH AT GOOD LEG OF COND BRANCH
                                ;18952
                                ;18953         ;1111-----------------------------;
                                ;18954         MDR_ZEXT(OSR),                  ; GET BRANCH DISPLACEMENT
U 007F, 0480,0036,4030,05E7,0072,3 ;18955      NEXT/CO.BRCND-BRANCH            ; FINNISH AT GOOD LEG OF COND BRANCH
                                ;18956
                                ;18957 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
```

```
                                        .TOC    "       Control Instructions            : BRW, JMP"

                                        ;******************************************************************************
                                        ;               BRW     displ.bw
                                        ;                  Input                    OSR                 Bits 7-0 of displ
                                        ;                                           XB                  Bits 15-8 of displ
                                        ;******************************************************************************
                                        .REGION/IRD1.R1L,IRD1.R1H
                                        =1000
                                        CO.BRW:
                                                ;1000---------------------------;
                                                MDR_ZEXT(OSR),                  ; GET LOW HALF OF DISPLACEMENT
                                                R[TEMP0]_ZEXT(XB) PC_PC+1,      ; GET HIGH HALF OF DISPLACEMENT
U 0098, 0485,759E,4000,25E7,0073,0              NEXT/CO.BRW-ALLIGN              ;

                                                ;1001---------------------------;
                                                MDR_ZEXT(OSR),                  ; GET LOW HALF OF DISPLACEMENT
                                                R[TEMP0]_ZEXT(XB) PC_PC+1,      ; GET HIGH HALF OF DISPLACEMENT
U 0099, 0485,759E,4000,25E7,0073,0              NEXT/CO.BRW-ALLIGN              ;

                                                ;1010---------------------------;
                                                MDR_ZEXT(OSR),                  ; GET LOW HALF OF DISPLACEMENT
                                                R[TEMP0]_ZEXT(XB) PC_PC+1,      ; GET HIGH HALF OF DISPLACEMENT
U 009A, 0485,759E,4000,25E7,0073,0              NEXT/CO.BRW-ALLIGN              ;

                                                ;1011---------------------------;
                                                MDR_ZEXT(OSR),                  ; GET LOW HALF OF DISPLACEMENT
                                                R[TEMP0]_ZEXT(XB) PC_PC+1,      ; GET HIGH HALF OF DISPLACEMENT
U 009B, 0485,759E,4000,25E7,0073,0              NEXT/CO.BRW-ALLIGN              ;

                                                ;1100---------------------------;
                                                MDR_ZEXT(OSR),                  ; GET LOW HALF OF DISPLACEMENT
                                                R[TEMP0]_ZEXT(XB) PC_PC+1,      ; GET HIGH HALF OF DISPLACEMENT
U 009C, 0485,759E,4000,25E7,0073,0              NEXT/CO.BRW-ALLIGN              ;

                                                ;1101---------------------------;
                                                MDR_ZEXT(OSR),                  ; GET LOW HALF OF DISPLACEMENT
                                                R[TEMP0]_ZEXT(XB) PC_PC+1,      ; GET HIGH HALF OF DISPLACEMENT
U 009D, 0485,759E,4000,25E7,0073,0              NEXT/CO.BRW-ALLIGN              ;

                                                ;1110---------------------------;
                                                MDR_ZEXT(OSR),                  ; GET LOW HALF OF DISPLACEMENT
                                                R[TEMP0]_ZEXT(XB) PC_PC+1,      ; GET HIGH HALF OF DISPLACEMENT
U 009E, 0485,759E,4000,25E7,0073,0              NEXT/CO.BRW-ALLIGN              ;
```

```
                                ;19002          ;1111-------------------------;
                                ;19003          MDR_ZEXT(OSR),                  ; GET LOW HALF OF DISPLACEMENT
                                ;19004          R[TEMP0]_ZEXT(XB) PC_PC+1,      ; GET HIGH HALF OF DISPLACEMENT
U 009F, 0485,759E,4000,25E7,0073,0 ;19005       NEXT/CO.BRW-ALLIGN              ;
                                ;19006 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                ;19007 CO.BRW-ALLIGN:
                                ;19008          ;--------------------------------;
                                ;19009          MDR_M[MDR].OR.(R[TEMP0].RR.24), ; CONCATINATE TWO HALVES OF DISP
U 0230, 0081,2292,4020,0467,0072,3 ;19010       NEXT/CO.BRCND-BRANCH            ; CONTINUE IN COND BRANCH FLOWS
                                ;19011
                                ;19012
                                ;19013 ;*************************************************************************************
                                ;19014 ;        JMP     dest.ab
                                ;19015 ;           Input                                MDR             Address to jump to
                                ;19016 ;                                                DSIZE LATCH     Long
                                ;19017 ;*************************************************************************************
                                ;19018 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;19019 CO.JMP:
                                ;19020          ;--------------------------------;
                                ;19021          PC_M[MDR],                      ; STUFF PC WITH NEW ADDRESS
U 0396, 0081,2002,403D,8487,0072,2 ;19022       NEXT/CO.NOP                     ; WASTE A CYCLE TO LET IDR1 CATCH PC
```

```
                                         .TOC     ''        Control Instructions              : BB''

                                         ;************************************************************************
                                         ;        BB(S,C,SS,CS,SC,CC,SSI,CCI)     pos.rl, base.ab, displ.bb
                                         ;           Input (Dest is in memory)    Q                Position
                                         ;                                        MDR              Base
                                         ;                                        DSIZE LATCH      Long
                                         ;                                        XB               Displ
                                         ;           Input (Dest is register)     MDR              Position
                                         ;                                        GPR(rnum)        Base
                                         ;                                        RNUM             Register number of base
                                         ;                                        DSIZE LATCH      Long
                                         ;                                        XB               Displ
                                         ;************************************************************************

                                         .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                         =1110    ;1110-----------------------------;
                                                  READ.LONG,                         ; GET WORD CONTAINING BIT TO TEST
                                                  Q_ZLITPL[1],                       ; MAKE MASK TO TEST BIT
U 00BE, 0180,0AF7,1670,0851,0072,6                IR<2-0>?,NEXT/CO.BB-MEM.IR0        ; IF = 0 TEST FOR BIT SET
                                                                                     ; IF = 1 TEST FOR BIT CLEAR
                                         CO.BBS:
                                         CO.BBC:
                                                  ;1111-----------------------------;
                                                  R[TEMP1]_M[MDR].ASR.3,             ; SHIFT AWAY BIT POINTER
U 00BF, 0C85,2777,0030,4047,04[illegible]         PUSH,NEXT/CO.BB-GET.BASE           ; CALL SUBROUTINE TO GET POSITION

                                         =1110    ;1110-----------------------------;
                                                  READ.LONG.MOD,                     ; GET WORD CONTAINING BIT TO TEST
                                                  Q_ZLITPL[1],                       ; MAKE MASK TO TEST BIT
U 00DE, 0980,0AF7,1670,0855,0072,6                IR<2-0>?,NEXT/CO.BB-MEM.IR0        ; IF = 0 TEST FOR BIT SET
                                                                                     ; IF = 1 TEST FOR BIT CLEAR
                                         CO.BBSS:
                                         CO.BBCS:
                                         CO.BBSC:
                                         CO.BBCC:
                                                  ;1111-----------------------------;
                                                  R[TEMP1]_M[MDR].ASR.3,             ; SHIFT AWAY BIT POINTER
U 00DF, 0C85,2777,0030,4047,0428,6                PUSH,NEXT/CO.BB-GET.BASE           ; CALL SUBROUTINE TO GET POSITION
                                         =1110
                                                  ;1110-----------------------------;
                                                  VA_M[VA].ANDNOT.ZLITO[3],          ; MASK OFF ADDRESS BITS 1-0
U 00FE, 0981,8C13,8030,1CA7,0073,2                NEXT/CO.BBI-READ                   ;

                                         CO.BBSSI:
                                         CO.BBCCI:
                                                  ;1111-----------------------------;
                                                  R[TEMP1]_M[MDR].ASR.3,             ; SHIFT AWAY BIT POINTER
U 00FF, 0C85,2777,0030,4047,0428,6                PUSH,NEXT/CO.BB-GET.BASE           ; CALL SUBROUTINE TO GET POSITION
```

```
                                    ;19072 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                    ;19073 CO.BBI-READ:
                                    ;19074          ;-------------------------------;
                                    ;19075          READ.MOD.LOCK,SIZE[LONG],       ; GET WORD CONTAINING BIT TO TEST
                                    ;19076          Q_ZLITPL[1],                    ; MAKE MASK TO TEST BIT
U 0732, 0080,0AF7,1660,0853,0072,6  ;19077          IR<2-0>?,NEXT/CO.BB-MEM.IR0     ; IF = 0 TEST FOR BIT SET
                                    ;19078
                                    ;19079 =110
                                    ;19080 CO.BB-MEM.IR0:
                                    ;19081          ;110-----------------------------;
                                    ;19082          WB_M[MDR].AND.Q,                ; TEST BIT FROM MEMORY FOR SET
U 07?6, 0C81,200A,0A70,0047,0072,4  ;19083          WX.NE.0?,NEXT/CO.BB-MEM.FIXPC   ; BRANCH IF WX NE 0
                                    ;19084
                                    ;19085          ;111-----------------------------;
                                    ;19086          WB_M[MDR].AND.Q,                ; TEST BIT FROM MEMORY FOR CLEAR
U 0727, 0881,200A,0A30,0047,0072,4  ;19087          WX.EQ.0?,NEXT/CO.BB-MEM.FIXPC   ; BRANCH IF WX EQ 0
                                    ;19088
                                    ;19089 =0
                                    ;19090 CO.BB-MEM.FIXPC:
                                    ;19091          ;0-------------------------------;
                                    ;19092          PC_PC+1 MB_XB,                  ; NO BRANCH SO SKIP OVER DISPLACEMENT
U 0724, 0081,7036,4640,2047,0072,8  ;19093          IR<2-0>?,NEXT/CO.BB-MEM.EXIT    ;
                                    ;19094
                                    ;19095          ;1-------------------------------;
                                    ;19096          PC_PC+SEXT(XB)+1,               ; DO THE BRANCH
                                    ;19097          SIZE[BYTE],                     ; GET ONE BYTE FROM XB
U 0725, 0081,7895,064D,A587,0072,8  ;19098          IR<2-0>?,NEXT/CO.BB-MEM.EXIT    ;
```

```
                                ;19099 =000
                                ;19100 CO.BB-MEM.EXIT:
                                ;19101          ;000-------------------------------; BBS
U 0728, 0080,0036,4130,0047,003F,9 ;19102       IRD1                              ; LET PC SETTLE BEFORE DOING IRD1
                                ;19103
                                ;19104          ;001-------------------------------; BBC
U 0729, 0080,0036,4130,0047,003F,9 ;19105       IRD1                              ; LET PC SETTLE BEFORE DOING IRD1
                                ;19106
                                ;19107          ;010-------------------------------; BBSS
                                ;19108          WRITE.LONG M[MDR].OR.Q,           ; CLEAR BIT TESTED
U 072A, 0481,200A,4120,0559,003F,9 ;19109       SIZE[LONG],IRD1                   ; WRITE OPERAND BACK TO MEMORY
                                ;19110
                                ;19111          ;011-------------------------------; BBCS
                                ;19112          WRITE.LONG M[MDR].OR.Q,           ; CLEAR BIT TESTED
U 072B, 0481,200A,4120,0559,003F,9 ;19113       SIZE[LONG],IRD1                   ; WRITE OPERAND BACK TO MEMORY
                                ;19114
                                ;19115          ;100-------------------------------; BBSC
                                ;19116          WRITE.LONG M[MDR].ANDNOT.Q,       ; SET BIT TESTED
U 072C, 0081,200B,8120,0559,003F,9 ;19117       SIZE[LONG],IRD1                   ; WRITE OPERAND BACK TO MEMORY
                                ;19118
                                ;19119          ;101-------------------------------; BBCC
                                ;19120          WRITE.LONG M[MDR].ANDNOT.Q,       ; SET BIT TESTED
U 072D, 0081,200B,8120,0559,003F,9 ;19121       SIZE[LONG],IRD1                   ; WRITE OPERAND BACK TO MEMORY
                                ;19122
                                ;19123          ;110-------------------------------; BBSSI
                                ;19124          WRITE.UL M[MDR].OR.Q,             ; WRITE DATA AND UNLOCK BUS
U 072E, 0C81,200A,4120,055B,003F,9 ;19125       SIZE[LONG],IRD1                   ;
                                ;19126
                                ;19127          ;111-------------------------------; BBCCI
                                ;19128          WRITE.UL M[MDR].ANDNOT.Q,         ; WRITE DATA AND UNLOCK BUS
U 072F, 0881,200B,8120,055B,003F,9 ;19129       SIZE[LONG],IRD1                   ;
                                ;19130
                                ;19131
                                ;19132 ;*****************************************************************************
                                ;19133 ;       SUBROUTINE TO CALL OS.VADD AND GET THE BASE OPERAND
                                ;19134 ;       IT ALSO FORMS THE ADDRESS OF THE BIT TO BE TESTED BY
                                ;19135 ;       ADDING THE BASE TO POSITION.ASR.3
                                ;19136 ;*****************************************************************************
                                ;19137
                                ;19138 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;19139 =0
                                ;19140 CO.BB-GET.BASE:
                                ;19141          ;0---------------------------------;
                                ;19142          PL_M[MDR],                        ; SAVE BIT POINTER IN PLATCH
                                ;19143          BRA ON ADD?,                      ; DON'T LOAD OSR WHEN GONIG TO OS.VADD
U 0286, 0C81,2BC2,463D,8047,0013,0 ;19144       NEXT/OS.VADD                      ; EVALUATE BASE OPERAND
                                ;19145
                                ;19146 CO.BB-RET:
                                ;19147          ;1---------------------------------; OS.VADD RETURNS HERE THROUGH ROM
                                ;19148          VA_M[MDR]+R[TEMP1],               ; LONG WORD ADD IS BASE+POS.ASR.3
                                ;19149          PL<4-3>_WB,                       ; SAVE BYTE POINTER IN PLATCH
U 0287, 0481,2FC1,00B0,44A7,03FF,F ;19150       RETURN [-1]                       ; RETURN -1
```

```
                                ;19151 CO.BB-REG:
                                ;19152          ;----------------------------------;
                                ;19153          WB_M[MDR].ANDNOT.ZLIT0[1F],       ; TEST TO SEE IF POS IS GEQ 32
U 0397, 0181,2C13,8A30,F847,0073,4 ;19154       WX.EQ.0?,NEXT/CO.BB-FAULT         ;
                                ;19155
                                ;19156 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                ;19157 =0
                                ;19158 CO.BB-FAULT:
                                ;19159          ;0----------------------------------;
U 0734, 0C80,0036,4030,0047,00FF,8 ;19160       NEXT/IE.OPER.FAULT                ; LET CHARLIE WORRY ABOUT IT
                                ;19161
                                ;19162          ;1----------------------------------;
U 0735, 0C81,2BC2,403D,8047,0075,0 ;19163       PL_M[MDR]                         ; PUT BIT COUNT INTO PLATCH
                                ;19164
                                ;19165          ;----------------------------------;
                                ;19166          0_ZLITPL[1],                      ; MAKE A MASK TO TEST BIT
U 0750, 0180,0AF7,1670,0847,0073,6 ;19167       IR<2-0>?,NEXT/CO.BB-REG.IR0       ; IF = 0 TEST FOR BIT SET
                                ;19168                                            ; IF = 1 TEST FOR BIT CLEAR
                                ;19169
                                ;19170 =110
                                ;19171 CO.BB-REG.IR0:
                                ;19172          ;110---------------------------------;
                                ;19173          WB_R[GPR.R].AND.Q,                ; TEST BIT FROM REGISTER FOR SET
U 0736, 0480,003A,0A7C,C047,0074,6 ;19174       WX.NE.0?,NEXT/CO.BB-REG.FIXPC     ; BRANCH IF WX NE 0
                                ;19175
                                ;19176          ;111---------------------------------;
                                ;19177          WB_R[GPR.R].AND.Q,                ; TEST BIT FROM REGISTER FOR CLEAR
U 0737, 0080,003A,0A3C,C047,0074,6 ;19178       WX.EQ.0?,NEXT/CO.BB-REG.FIXPC     ; BRANCH IF WX EQ 0
                                ;19179
                                ;19180 =0
                                ;19181 CO.BB-REG.FIXPC:
                                ;19182          ;0----------------------------------;
                                ;19183          PC_PC+1 MB_XB,                    ; NO BRANCH SO SKIP OVER DISPLACEMENT
U 0746, 0881,7036,4640,2047,0073,8 ;19184       IR<2-0>?,NEXT/CO.BB-REG.EXIT      ;
                                ;19185
                                ;19186          ;1----------------------------------;
                                ;19187          PC_PC+SEXT(XB)+1,                 ; DO THE BRANCH
                                ;19188          SIZE[BYTE],                       ; GET ONE BYTE FROM XB
U 0747, 0881,7895,064D,A587,0073,8 ;19189       IR<2-0>?,NEXT/CO.BB-REG.EXIT      ;
```

```
                                ;19190 =000
                                ;19191 CO.BB-REG.EXIT:
                                ;19192          ;000-----------------------------; BBS
U 0738, 0080,0036,4130,0047,003F,9 ;19193       IRD1                              ; LET PC SETTLE BEFORE DOING IRD1
                                ;19194
                                ;19195          ;001-----------------------------; BBC
U 0739, 0080,0036,4130,0047,003F,9 ;19196       IRD1                              ; LET PC SETTLE BEFORE DOING IRD1
                                ;19197
                                ;19198          ;010-----------------------------; BBSS
                                ;19199          R[GPR.R]_RB.OR.Q,                 ; CLEAR BIT TESTED
U 073A, 0484,003A,413C,C047,003F,9 ;19200       IRD1                              ; WRITE OPERAND BACK TO REGISTER
                                ;19201
                                ;19202          ;011-----------------------------; BBCS
                                ;19203          R[GPR.R]_RB.OR.Q,                 ; CLEAR BIT TESTED
U 073B, 0484,003A,413C,C047,003F,9 ;19204       IRD1                              ; WRITE OPERAND BACK TO REGISTER
                                ;19205
                                ;19206          ;100-----------------------------; BBSC
                                ;19207          R[GPR.R]_RB.ANDNOT.Q,             ; SET BIT TESTED
U 073C, 0084,003B,813C,C047,003F,9 ;19208       IRD1                              ; WRITE OPERAND BACK TO REGISTER
                                ;19209
                                ;19210          ;101-----------------------------; BBCC
                                ;19211          R[GPR.R]_RB.ANDNOT.Q,             ; SET BIT TESTED
U 073D, 0084,003B,813C,C047,003F,9 ;19212       IRD1                              ; WRITE OPERAND BACK TO REGISTER
                                ;19213
                                ;19214          ;110-----------------------------; BBSSI
                                ;19215          R[GPR.R]_RB.OR.Q,                 ; WRITE DATA
U 073E, 0484,003A,413C,C047,003F,9 ;19216       IRD1                              ;
                                ;19217
                                ;19218          ;111-----------------------------; BBCCI
                                ;19219          R[GPR.R]_RB.ANDNOT.Q,             ; WRITE DATA
U 073F, 0084,003B,813C,C047,003F,9 ;19220       IRD1                              ;
```

```
                                ;19221 .TOC   "       Control Instructions            : BLB"
                                ;19222
                                ;19223 ;*****************************************************************************
                                ;19224 ;       BLB(S,C) src.rl,displ.bb
                                ;19225 ;          Input (Src is memory)        MDR             Src
                                ;19226 ;                                       XB              Displ
                                ;19227 ;
                                ;19228 ;          Input (Src is register)      GPR(rnum)       Src
                                ;19229 ;                                       RNUM            Register number of src
                                ;19230 ;                                       XB              Displ
                                ;19231 ;*****************************************************************************
                                ;19232
                                ;19233 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;19234 CO.BLBS:
                                ;19235         ;-------------------------------;
                                ;19236         WB_M[MDR].AND.2LIT0[1],         ; TEST BIT FOR SET
U 039D, 0581,2C12,0A30,0847,0074,E ;19237      WX.EQ.0?,NEXT/CO.BLB-TEST       ;
                                ;19238
                                ;19239 CO.BLBC:
                                ;19240         ;-------------------------------;
                                ;19241         WB_M[MDR].AND.2LIT0[1],         ; TEST BIT FOR SET
U 039E, 0181,2C12,0A70,0847,0074,E ;19242      WX.NE.0?,NEXT/CO.BLB-TEST       ;
                                ;19243
                                ;19244 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                ;19245 =0
                                ;19246 CO.BLB-TEST:
                                ;19247         ;0------------------------------;
                                ;19248         PC_PC+SEXT(XB)+1,               ; DO THE BRANCH
                                ;19249         SIZE[BYTE],                     ; DISP IS A BYTE
U 074E, 0481,7895,000D,A587,0072,2 ;19250      NEXT/CO.NOP                     ; GO WASTE SOME TIME
                                ;19251
                                ;19252         ;1------------------------------;
                                ;19253         PC_PC+1 MB_XB,                  ; NO BRANCH SO SKIP OVER DISPLACEMENT
U 074F, 0481,7036,4000,2047,0072,2 ;19254      NEXT/CO.NOP                     ; GO WASTE SOME TIME
```

```
                    ;19255 .TOC   "       Control Instructions            : ACB"
                    ;19256
                    ;19257 ;*************************************************************************
                    ;19258 ;       ACB(B, W, L)    limit.rx, add.rx, index.rx, displ.bw
                    ;19259 ;          Input (Index is memory)      Q                 Limit
                    ;19260 ;                                       MDR               Add
                    ;19261 ;                                       OSR               OSR of index
                    ;19262 ;                                       DSIZE LATCH       Size of index
                    ;19263 ;                                       XB                Displ
                    ;19264 ;          Input (Index is register)    Q                 Limit
                    ;19265 ;                                       MDR               Add
                    ;19266 ;                                       GPR(rnum)         Index
                    ;19267 ;                                       RNUM              Register number of index
                    ;19268 ;                                       DSIZE LATCH       Size of index
                    ;19269 ;                                       XB                Displ
                    ;19270 ;*************************************************************************
                    ;19271 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                    ;19272 =00000
                    ;19273 =01111
                    ;19274 CO.ACBB-MEM:
                    ;19275 CO.ACBW-MEM:
                    ;19276 CO.ACBL-MEM:
                    ;19277          ;01111----------------------------;
                    ;19278          M[TEMP1]_Q Q_D,                   ; SAVE LIMIT IN RTEMP1
                    ;19279          PUSH,BRA ON ADD?,                 ; CALL OS.MOD
U 028F, 0C86,102C,7630,0047,0411,0  ;19280          NEXT/OS.MOD                       ; RETURN WILL BE +1
                    ;19281
                    ;19282          ;10000----------------------------;
                    ;19283          D_SEXT(XB) PC_PC+2,               ; MAKE SURE WE CAN ACCESS THE DISP
U 0290, 0481,7816,689D,A047,0075,C  ;19284          IR<2>?,NEXT/CO.ACB-SETCC          ; BRANCH ON IR2  (ACBB,ACBW) OR (ACBL)
                    ;19285 =
                    ;19286 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                    ;19287 =0
                    ;19288 CO.ACB-SETCC:
                    ;19289          ;0-------------------------------; HERE ON ACBL
                    ;19290          MTEMP0_MDR+Q,                     ; WBUS AND MTEMP0 GET NEW INDEX
                    ;19291          WRITE,SIZE[IDEP],                 ; WRITE NEW INDEX BACK TO MEMORY
                    ;19292          CCOP2,                            ; SET CONDITION CODES
U 075C, 0487,2009,0030,15D8,0075,2  ;19293          NEXT/CO.ACB-TEST.ADDEND           ;
                    ;19294
                    ;19295          ;1-------------------------------; HERE ON ACBB AND ACBW
                    ;19296          MTEMP0_MDR+Q,                     ; WBUS AND MTEMP0 GET NEW INDEX
                    ;19297          WRITE,SIZE[IDEP],                 ; WRITE NEW INDEX BACK TO MEMORY
                    ;19298          CCOP1,                            ; SET CONDITION CODES
U 075D, 0487,2009,0030,0DD8,0075,2  ;19299          NEXT/CO.ACB-TEST.ADDEND           ;
```

```
                                ;19300 CO.ACB-TEST.ADDEND:
                                ;19301          ;-------------------------------;
                                ;19302          WB_Q,                           ; DRIVE ADDEND THROUGH ALU FOR TEST
                                ;19303          SIGND CMP?,SIZE[IDEP],          ; TEST FOR GEQ 0
U 0752, 0480,003A,4B7D,8047,0073,1  ;19304      NEXT/CO.ACB-TEST1               ;
                                ;19305
                                ;19306 =01
                                ;19307 CO.ACB-TEST1:
                                ;19308          ;01-----------------------------; HERE FOR ADDEND GEQ 0
                                ;19309          WB_R[TEMP1]-M[TEMP0],           ; WB <- LIMIT-INDEX
                                ;19310          SIGND CMP?,SIZE[IDEP],          ; TEST FOR INDEX LEQ LIMIT
U 0731, 0C80,0003,0B70,4047,0875,1 374*  ;19311 NEXT/CO.ACB-TEST2               ;
                                ;19312
                                ;19313          ;11-----------------------------; HERE FOR ADDEND LSS 0
                                ;19314          WB_M[TEMP0]-R[TEMP1],           ; WB <- INDEX-LIMIT
                                ;19315          SIGND CMP?,SIZE[IDEP],          ; TEST FOR INDEX GEQ LIMIT
U 0733, 0C80,0000,0B70,4047,0875,1 374*  ;19316 NEXT/CO.ACB-TEST2               ;
                                ;19317
                                ;19318 =01
                                ;19319 CO.ACB-TEST2:
                                ;19320          ;01-----------------------------;
                                ;19321          PC_PC+D,                        ; DO THE BRANCH
U 0751, 0C80,0022,403D,8587,0072,2  ;19322      NEXT/CO.NOP                     ; WASTE A LITTLE TIME
                                ;19323
                                ;19324          ;11-----------------------------;
U 0753, 0080,0036,4130,0047,003F,9  ;19325      IRD1                            ; NOTHING TO DO
```

```
                                ;19326 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;19327 CO.ACB-REG:
                                ;19328          ;--------------------------------;
                                ;19329          D_SEXT(XB) PC_PC+2,             ; MAKE SURE WE CAN ACCESS THE DISP
U 039F, 0C81,7816,689D,A047,0028,C ;19330       IR<2>?                          ; BRANCH ON OPCODE
                                ;19331
                                ;19332 =0
                                ;19333 CO.ACBL-REG:
                                ;19334          ;0-------------------------------;
                                ;19335          R[GPR.R]_M[MDR]+RB,             ; WRITE NEW INDEX TO GPR[RNUM]
                                ;19336          CCOP2,SIZE[IDEP],               ; SET CONDITION CODES
U 028C, 0485,2001,003C,D047,0075,4 ;19337       NEXT/CO.ACB-TEST.ADD.REG        ;
                                ;19338 CO.ACBB-REG:
                                ;19339 CO.ACBW-REG:
                                ;19340          ;1-------------------------------;
                                ;19341          R[GPR.R].SIZ_M[MDR]+RB,         ; WRITE NEW INDEX TO GPR[RNUM]
                                ;19342          CCOP1,SIZE[IDEP],               ; SET CONDITION CODES
U 028D, 0483,2001,003C,C847,0075,4 ;19343       NEXT/CO.ACB-TEST.ADD.REG        ;
                                ;19344
                                ;19345 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                ;19346 CO.ACB-TEST.ADD.REG:
                                ;19347          ;--------------------------------;
                                ;19348          WB_M[MDR],                      ; DRIVE ADDEND THROUGH ALU FOR TEST
U 0754, 0081,2592,4B70,0047,0875,5 351* ;19349  SIGND CMP?,SIZE[IDEP]            ; TEST FOR GEQ 0
                                ;19350
                                ;19351 =01      ;01-------------------------------;
                                ;19352          WB_Q-R[GPR.R],                  ; WB <- LIMIT-INDEX
                                ;19353          SIGND CMP?,SIZE[IDEP],          ; TEST FOR INDEX LEQ LIMIT
U 0755, 0C80,003B,0B7C,C047,0875,1 436* ;19354  NEXT/CO.ACB-TEST2                ;
                                ;19355
                                ;19356          ;11-------------------------------;
                                ;19357          WB_R[GPR.R]-Q,                  ; WB <- INDEX-LIMIT
                                ;19358          SIGND CMP?,SIZE[IDEP],          ; TEST FOR INDEX GEQ LIMIT
U 0757, 0C80,0038,0B7C,C047,0875,1 436* ;19359  NEXT/CO.ACB-TEST2                ;
```

```
                        ;19360 .TOC     "       Control Instructions            : AOB"
                        ;19361
                        ;19362 ;*************************************************************************
                        ;19363 ;       AOBL(SS,EQ)     limit.rl, index.ml, displ.bb
                        ;19364 ;          Input (Index is memory)      Q                 Limit
                        ;19365 ;                                       MDR               Index
                        ;19366 ;                                       VA                Address of index
                        ;19367 ;                                       DSIZE LATCH       Long
                        ;19368 ;                                       XB                Displ
                        ;19369 ;          Input (Index is register)
                        ;19370 ;                                       MDR               Limit
                        ;19371 ;                                       GPR(rnum)         Index
                        ;19372 ;                                       RNUM              Register number of index
                        ;19373 ;                                       DSIZE LATCH       Long
                        ;19374 ;                                       XB                Displ
                        ;19375 ;*************************************************************************
                        ;19376
                        ;19377 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;19378 CO.AOBLSS:
                        ;19379 CO.AOBLEQ:
                        ;19380         ;-------------------------------;
                        ;19381         D_SEXT(XB) PC_PC+1,             ; MAKE SURE WE HAVE ACCESS TO DISP
U 03A1, 0881,7816,690D,A047,0075,E      ;19382         REG MODE?,NEXT/CO.AOB-REG.TEST  ; INDEX IN A GPR ???
                        ;19383
                        ;19384 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                        ;19385 =0
                        ;19386 CO.AOB-REG.TEST:
                        ;19387         ;0-------------------------------;
                        ;19388         MTEMP0_MDR+1,                   ; UPDATE THE INDEX
                        ;19389         WRITE,SIZE[IDEP],               ; WRITE IT BACK TO MEMORY
                        ;19390         CCOP2,                          ; SET CONDITION CODES
U 075E, 0087,2E50,0030,15D8,0075,B      ;19391         NEXT/CO.AOB-MEM                 ;
                        ;19392
                        ;19393         ;1-------------------------------;
                        ;19394         R[GPR.R]_RB+1,                  ; UPDATE THE INDEX
U 075F, 0884,0E7C,003C,D047,0075,6      ;19395         CCOP2,SIZE[IDEP]                ; SET CONDITION CODES
                        ;19396
                        ;19397         ;-------------------------------;
                        ;19398         WB_M[MDR]-R[GPR.R],             ; WB <- LIMIT - INDEX
                        ;19399         SIGND CMP .NOT.IRO?,            ; TEST SIGN AND OPCODE
U 0756, 0881,2000,0D7C,C047,0874,0 380* ;19400         SIZE[IDEP],NEXT/CO.AOB-BRA      ;
                        ;19401
                        ;19402 CO.AOB-MEM:
                        ;19403         ;-------------------------------;
                        ;19404         WB_Q-M[TEMP0],SIZE[IDEP],       ; WB <- LIMIT - INDEX
U 075B, 0C80,000B,0D70,0047,0874,0 373* ;19405         SIGND CMP .NOT.IRO?             ; BRANCH ON RESULT AND OPCODE
```

```
                                ;19406 =000
                                ;19407 CO.AOB-BRA:
                                ;19408          ;000-----------------------------; AOBLEQ
                                ;19409          PC_PC+D,                          ; DO THE BRANCH
U 0740, 0C80,0022,403D,8587,0072,2  ;19410      NEXT/CO.NOP                       ;
                                ;19411
                                ;19412          ;001-----------------------------; AOBLSS
                                ;19413          PC_PC+D,                          ; DO THE BRANCH
U 0741, 0C80,0022,403D,8587,0072,2  ;19414      NEXT/CO.NOP                       ;
                                ;19415
                                ;19416          ;010-----------------------------; AOBLEQ
                                ;19417          PC_PC+D,                          ; DO THE BRANCH
U 0742, 0C80,0022,403D,8587,0072,2  ;19418      NEXT/CO.NOP                       ;
                                ;19419
                                ;19420          ;011-----------------------------; AOBLSS
U 0743, 0080,0036,4130,0047,003F,9  ;19421      IRD1                              ; NO BRANCH
                                ;19422
                                ;19423          ;100-----------------------------; AOBLEQ
U 0744, 0080,0036,4130,0047,003F,9  ;19424      IRD1                              ; NO BRANCH
                                ;19425
                                ;19426          ;101-----------------------------; AOBLSS
U 0745, 0080,0036,4130,0047,003F,9  ;19427      IRD1                              ; NO BRANCH
                                ;19428 =
```

```
                                ;19429 .TOC    "       Control Instructions            : SOB"
                                ;19430
                                ;19431 ;*****************************************************************************
                                ;19432 ;       SOBG(EQ,TR)     index.ml, displ.bb
                                ;19433 ;          Input (Index is memory)      MDR             Index
                                ;19434 ;                                       VA              Address of index
                                ;19435 ;                                       XB              Displ
                                ;19436 ;          Input (Index is register)    GPR(rnum)       Index
                                ;19437 ;                                       RNUM            Register number of index
                                ;19438 ;                                       XB              Displ
                                ;19439 ;*****************************************************************************
                                ;19440 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;19441 CO.SOBGEQ:
                                ;19442 CO.SOBGTR:
                                ;19443         ;-----------    ----------------;
U 03A2, 0081,7816,500D,A047,0076,6      ;19444         Q_SEXT(XB) ;...C+1              ; MAKE SURE WE HAVE ACCESS TO DISP
                                ;19445
                                ;19446 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                ;19447         ;-------------------------------; INDEX IS IN MEMORY
                                ;19448         R[DST.R]_D_M[MDR]-1,            ; SUBTRACT ONE ...
                                ;19449         WRITE NOTREG,SIZE[IDEP],        ; WRITE INDEX BACK TO MEMORY
                                ;19450         CCOP2 SIGND CMP .NOT.IR0?,      ; BRANCH ON TEST AND OPCODE
U 0766, 0885,2E51,2D7C,55DA,0874,8 407* ;19451         NEXT/CO.SOB-BRA                 ;
                                ;19452
                                ;19453 CO.SOB-REG:
                                ;19454         ;-------------------------------; INDEX IS IN MEMORY
                                ;19455         R[GPR.R]_D_RB-1,SIZE[IDEP],     ; SUBTRACT ONE ...
U 0767, 0084,0E7D,2D7C,D047,0874,8 407* ;19456         CCOP2 SIGND CMP .NOT.IR0?       ; BRANCH ON TEST AND OPCODE
                                ;19457 =000
                                ;19458 CO.SOB-BRA:
                                ;19459         ;000----------------------------; SOBGTR
                                ;19460         PC_PC+Q,                        ; TAKE THE BRANCH
U 0748, 0C80,003A,403D,8587,0072,2      ;19461         NEXT/CO.NOP                     ; WASTE A LITTLE TIME
                                ;19462
                                ;19463         ;001----------------------------; SOBGEQ
                                ;19464         PC_PC+Q,                        ; TAKE THE BRANCH
U 0749, 0C80,003A,403D,8587,0072,2      ;19465         NEXT/CO.NOP                     ; WASTE A LITTLE TIME
                                ;19466
                                ;19467         ;010----------------------------; SOBGTR (NO BRANCH PROBABLY???)
                                ;19468         WB_D-OLIT24[17F],               ; TEST FOR MOST POSITIVE CASE
                                ;19469         WX.EQ.0?,                       ; TEST FOR 7FFFFFFF
U 074A, 0580,0EB0,0A3B,F847,0876,0 378* ;19470         NEXT/CO.SOB-TEST.MOST.POS       ;
                                ;19471
                                ;19472         ;011----------------------------; SOBGEQ
                                ;19473         PC_PC+Q,                        ; TAKE THE BRANCH
U 074B, 0C80,003A,403D,8587,0072,2      ;19474         NEXT/CO.NOP                     ; WASTE A LITTLE TIME
                                ;19475
                                ;19476         ;100----------------------------; SOBCTR (NO BRANCH PROBABLY???)
                                ;19477         WB_D-OLIT24[17F],               ; TEST FOR MOST POSITIVE CASE
                                ;19478         WX.EQ.0?,                       ; TEST FOR 7FFFFFFF
U 074C, 0580,0EB0,0A3B,F847,0876,0 378* ;19479         NEXT/CO.SOB-TEST.MOST.POS       ;
                                ;19480
                                ;19481         ;101----------------------------; SOBGEQ (NO BRANCH PROBABLY???)
                                ;19482         WB_D-OLIT24[17F],               ; TEST FOR MOST POSITIVE CASE
                                ;19483         WX.EQ.0?,                       ; TEST FOR 7FFFFFFF
```

```
U 074D, 0580,0EB0,0A3B,F847,0876,0 378*  ;19484          NEXT/CO.SOB-TEST.MOST.POS       ;
                                         ;19485 =
                                         ;19486 =0
                                         ;19487 CO.SOB-TEST.MOST.POS:
                                         ;19488          ;0------------------------------;
U 0760, 0080,0036,4130,0047,003F,9       ;19489          IRD1                            ; ALL OK
                                         ;19490
                                         ;19491          ;1------------------------------;
                                         ;19492          PC PC+Q,                        ; TAKE THE BRANCH
U 0761, 0C80,003A,403D,8587,0072,2       ;19493          NEXT/CO.NOP                     ; WASTE A LITTLE TIME
```

```
                                                                I  5
;19494 .TOC    "       Control Instructions            : CASE"
;19495
;19496 ;*************************************************************************
;19497 ;       CASE(B,W,L)     selector.rx, base.rx, limit.rx, displ[0].bw,...displ[limit].
;19498 ;          Input (Limit is memory)      Q               Selector
;19499 ;                                       MDR             Base
;19500 ;                                       DSIZE LATCH     Size of limit
;19501 ;          Input (Limit is register)    Q               Selector
;19502 ;                                       MDR             Base
;19503 ;                                       DSIZE LATCH     Size of limit
;19504 ;                                       GPR(rnum)       Limit
;19505 ;                                       RNUM            Register number of limit
;19506 ;*************************************************************************
;19507 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
;19508 02AF:
;19509 CO.CASEB-MEM:
;19510 CO.CASEW-MEM:
;19511 CO.CASEL-MEM:
;19512         ;01111-----------------------------;
;19513         R[TEMP3]_Q-M[MDR],                  ; MTEMP3 <- SELECTOR-BASE
;19514         PUSH,BRA ON ADD?,                   ; CALL OS.RED TO GET LIMIT
U 02AF, 0085,200B,0630,C047,0410,0  ;19515         NEXT/OS.RED                         ;
;19516
;19517 2B0:    ;10000---------------------------;
;19518         WB_R[TEMP3]-M[MDR],                 ; COMPARE BY    (SELECTOR-BASE) - LIMIT
U 02B0, 0C81,2003,0030,D0C7,0076,8  ;19519         SIZE[IDEP].ALUS_UNSGN               ; LEAVE RESULT IN STATE LATCHES
;19520
;19521 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
;19522 CO.CASE-DISP.ADD:
;19523         ;-------------------------------;
;19524         R[TEMP4]_ZEXT(M[TEMP3]).SL.1,       ; FORM POINTER TO DISPLACEMENT
U 0768, 0C84,359D,C031,0047,0076,9  ;19525         SIZE[IDEP]                          ;
;19526
;19527         ;-------------------------------;
;19528         VA_M[PC]+R[TEMP4],                  ; GET ADDRESS OF DISPLACEMENT
U 0769, 0481,A001,0B71,1CA7,0075,8  ;19529         ALUS?,NEXT/CO.CASE-DECIDE           ; USE ALUS TO DECIDE WHAT TO DO
;19530 =00
;19531 CO.CASE-DECIDE:
;19532         ;00-------------------------------; NOT LESS THAN OR EQUAL
;19533         R[TEMP1]_M[MDR],                    ; SAVE LIMIT TO SET CC'S WITH
U 0758, 0085,2592,4930,4047,0076,2  ;19534         REG MODE?,NEXT/CO.CASE-BAD          ;
;19535
;19536         ;01-------------------------------; LESS THAN OR EQUAL
;19537         READ,                               ; GET DISPLACEMENT
;19538         SIZE[WORD],                         ; SIZE OF DISP IS ALWAYS WORD
;19539         R[TEMP1]_M[MDR],                    ; SAVE LIMIT IN TEMP1
U 0759, 0485,2592,4010,4050,0076,8  ;19540         NEXT/CO.CASE-GOOD                   ;
;19541
;19542         ;10-------------------------------; LESS THAN OR EQUAL
;19543         READ,                               ; GET DISPLACEMENT
;19544         SIZE[WORD],                         ; SIZE OF DISP IS ALWAYS WORD
;19545         R[TEMP1]_M[MDR],                    ; SAVE LIMIT IN TEMP1
U 075A, 0485,2592,4010,4050,0076,8  ;19546         NEXT/CO.CASE-GOOD                   ;
;19547 =
```

```
                                ;19548 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;19549 CO.CASEB-REG:
                                ;19550 CO.CASEW-REG:
                                ;19551 CO.CASEL-REG:
                                ;19552          ;--------------------------------;
U 03A3, 0885,200B,0030,C047,0076,A ;19553       R[TEMP3]_Q-M[MDR]                ; TEMP <- SELECTOR-BASE
                                ;19554
                                ;19555 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                ;19556          ;--------------------------------;
                                ;19557          WB_M[TEMP3]-R[GPR.R],            ; COMPARE BY   (SELECTOR-BASE) - LIMIT
                                ;19558          ALUS_UNSGN,SIZE[IDEP],           ; LEAVE RESULT IN STATE LATCHES
U 076A, 0C80,3000,003C,D0C7,0076,8 ;19559       NEXT?CO.CASE-DISP.ADD            ;
                                ;19560
                                ;19561 CO.CASE-GOOD:
                                ;19562 CO.CASE-BRANCH:
                                ;19563          ;--------------------------------;
                                ;19564          PC_PC+SEXT(M[MDR]),              ; ADD DISPLACEMENT TO PC
                                ;19565          SIZE[WORD],                      ; ONE WORD DISPLACEMENT FOR THIS ONE
U 076B, 0081,2816,491D,8587,0076,4 ;19566       REG MODE?,NEXT/CO.CASE-SETCC     ; GO SET CC'S WHILE PC SETTLES
                                ;19567
                                ;19568 =0
                                ;19569 CO.CASE-BAD:
                                ;19570          ;0-------------------------------; MEMORY MODE
                                ;19571          D_ZEXT(M[MDR]),SIZE[IDEP],       ; ZERO EXTEND LIMIT
U 0762, 0481,2016,603D,8047,0076,D ;19572       NEXT/CO.CASE-BAD.FIXPC           ;
                                ;19573
                                ;19574          ;1-------------------------------; REGISTER MODE
U 0763, 0886,25BE,403C,C047,0076,C ;19575       M[TEMP2]_R[GPR.R]                ; GET REGISTER TO MTEMP TO ZEXT
                                ;19576
                                ;19577          ;--------------------------------;
U 076C, 0080,2016,603D,8047,0076,D ;19578       D_ZEXT(M[TEMP2]),SIZE[IDEP]      ; ZERO EXTEND LIMIT
                                ;19579
                                ;19580 CO.CASE-BAD.FIXPC:
                                ;19581          ;--------------------------------;
                                ;19582          PC_PC+((D+ZLIT0[1]).SL.1),       ; PUSH PC PAST DISPATCH TABLE
U 076D, 0980,0C31,C930,0D87,0076,4 ;19583       REG MODE?,NEXT/CO.CASE-SETCC     ; GO SET CC'S WHILE PC SETTLES
                                ;19584
                                ;19585 =0
                                ;19586 CO.CASE-SETCC:
                                ;19587          ;0-------------------------------; MEMORY MODE
                                ;19588          WB_R[TEMP3]-M[TEMP1],            ; WB <- (SELECTOR-BASE) - LIMIT
                                ;19589          CCOP1,SIZE[IDEP],                ; SET CONDITION CODES
U 0764, 0480,1003,0130,C847,003F,9 ;19590       IRD1                             ; EXIT !!
                                ;19591
                                ;19592          ;1-------------------------------; REGISTER MODE
                                ;19593          WB_M[TEMP3]-R[GPR.R],            ; WB <- (SELECTOR-BASE) - LIMIT
                                ;19594          CCOP1,SIZE[IDEP],                ; SET CONDITION CODES
U 0765, 0080,3000,013C,C847,003F,9 ;19595       IRD1                             ; EXIT !!
```

```
                                ;19596 .TOC   ''      Control Instructions          : BSBB''
                                ;19597
                                ;19598 ;*************************************************************************
                                ;19599 ;       BSBB    displ.bx
                                ;19600 ;          Input                          XB              Displ
                                ;19601 ;*************************************************************************
                                ;19602 .REGION/IRD1.R1L,IRD1.R1H
                                ;19603 =1000
                                ;19604 CO.BSBB:
                                ;19605         ;1000-------------------------;
                                ;19606         MDR_ZEXT(OSR),                  ; GET DISPLACEMENT FROM OSR
U 01F8, 0085,A592,4030,05E7,0076,E ;19607      R[TEMP0]_M[PC],NEXT/CO.BSB      ; READ PC FOR LATER PUSH
                                ;19608
                                ;19609         ;1001-------------------------;
                                ;19610         MDR_ZEXT(OSR),                  ; GET DISPLACEMENT FROM OSR
U 01F9, 0085,A592,4030,05E7,0076,E ;19611      R[TEMP0]_M[PC],NEXT/CO.BSB      ; READ PC FOR LATER PUSH
                                ;19612
                                ;19613         ;1010-------------------------;
                                ;19614         MDR_ZEXT(OSR),                  ; GET DISPLACEMENT FROM OSR
U 01FA, 0085,A592,4030,05E7,0076,E ;19615      R[TEMP0]_M[PC],NEXT/CO.BSB      ; READ PC FOR LATER PUSH
                                ;19616
                                ;19617         ;1011-------------------------;
                                ;19618         MDR_ZEXT(OSR),                  ; GET DISPLACEMENT FROM OSR
U 01FB, 0085,A592,4030,05E7,0076,E ;19619      R[TEMP0]_M[PC],NEXT/CO.BSB      ; READ PC FOR LATER PUSH
                                ;19620
                                ;19621         ;1100-------------------------;
                                ;19622         MDR_ZEXT(OSR),                  ; GET DISPLACEMENT FROM OSR
U 01FC, 0085,A592,4030,05E7,0076,E ;19623      R[TEMP0]_M[PC],NEXT/CO.BSB      ; READ PC FOR LATER PUSH
                                ;19624
                                ;19625         ;1101-------------------------;
                                ;19626         MDR_ZEXT(OSR),                  ; GET DISPLACEMENT FROM OSR
U 01FD, 0085,A592,4030,05E7,0076,E ;19627      R[TEMP0]_M[PC],NEXT/CO.BSB      ; READ PC FOR LATER PUSH
                                ;19628
                                ;19629         ;1110-------------------------;
                                ;19630         MDR_ZEXT(OSR),                  ; GET DISPLACEMENT FROM OSR
U 01FE, 0085,A592,4030,05E7,0076,E ;19631      R[TEMP0]_M[PC],NEXT/CO.BSB      ; READ PC FOR LATER PUSH
                                ;19632
                                ;19633         ;1111-------------------------;
                                ;19634         MDR_ZEXT(OSR),                  ; GET DISPLACEMENT FROM OSR
U 01FF, 0085,A592,4030,05E7,0076,E ;19635      R[TEMP0]_M[PC],NEXT/CO.BSB      ; READ PC FOR LATER PUSH
                                ;19636 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                ;19637 CO.BSB:
                                ;19638         ;-----------------------------;
                                ;19639         VA_R[SP]_RB-CONX(4),            ; PUT ADDRESS POINTED TO BY SP INTO VA
U 076E, 0C85,473C,0027,84A7,0076,F ;19640      PUSH RBS-                       ; MAKE RESTART POSSIBLE IF WE DIE
                                ;19641
                                ;19642         ;-----------------------------;
                                ;19643         WRITE M[TEMP0],                 ; PUSH SAVED PC ONTO STACK
                                ;19644         SIZE[LONG],                     ; SIZE OF ADDRESSES IS LONG
U 076F, 0480,0592,4020,05D8,0072,3 ;19645      NEXT/CO.BRCND-BRANCH            ; FINNISH LIKE A REGULAR BRANCH
```

```
                                ;19646 .TOC   ''      Control Instructions          : BSBW''
                                ;19647
                                ;19648 ;*************************************************************************
                                ;19649 ;       BSBW    displ.bx
                                ;19650 ;          Input                          XB              Bits 16-8 of displ
                                ;19651 ;                                         OSR             Bits 7-0 of displ
                                ;19652 ;*************************************************************************
                                ;19653 .REGION/IRD1.R1L,IRD1.R1H
                                ;19654 =1000
                                ;19655 CO.BSBW:
                                ;19656         ;1000-------------------------;
                                ;19657         MDR_ZEXT(OSR),                ; GET DISPLACEMENT FROM OSR
                                ;19658         R[TEMP0]_ZEXT(XB) PC_PC+1,    ; GET HIGH HALF OF DISPLACEMENT
U 0218, 0C85,759E,4000,25E7,0077,0 ;19659      NEXT/CO.BSBW-ALLIGN           ;
                                ;19660
                                ;19661         ;1001-------------------------;
                                ;19662         MDR_ZEXT(OSR),                ; GET DISPLACEMENT FROM OSR
                                ;19663         R[TEMP0]_ZEXT(XB) PC_PC+1,    ; GET HIGH HALF OF DISPLACEMENT
U 0219, 0C85,759E,4000,25E7,0077,0 ;19664      NEXT/CO.BSBW-ALLIGN           ;
                                ;19665
                                ;19666         ;1010-------------------------;
                                ;19667         MDR_ZEXT(OSR),                ; GET DISPLACEMENT FROM OSR
                                ;19668         R[TEMP0]_ZEXT(XB) PC_PC+1,    ; GET HIGH HALF OF DISPLACEMENT
U 021A, 0C85,759E,4000,25E7,0077,0 ;19669      NEXT/CO.BSBW-ALLIGN           ;
                                ;19670
                                ;19671         ;1011-------------------------;
                                ;19672         MDR_ZEXT(OSR),                ; GET DISPLACEMENT FROM OSR
                                ;19673         R[TEMP0]_ZEXT(XB) PC_PC+1,    ; GET HIGH HALF OF DISPLACEMENT
U 021B, 0C85,759E,4000,25E7,0077,0 ;19674      NEXT/CO.BSBW-ALLIGN           ;
                                ;19675
                                ;19676         ;1100-------------------------;
                                ;19677         MDR_ZEXT(OSR),                ; GET DISPLACEMENT FROM OSR
                                ;19678         R[TEMP0]_ZEXT(XB) PC_PC+1,    ; GET HIGH HALF OF DISPLACEMENT
U 021C, 0C85,759E,4000,25E7,0077,0 ;19679      NEXT/CO.BSBW-ALLIGN           ;
                                ;19680
                                ;19681         ;1101-------------------------;
                                ;19682         MDR_ZEXT(OSR),                ; GET DISPLACEMENT FROM OSR
                                ;19683         R[TEMP0]_ZEXT(XB) PC_PC+1,    ; GET HIGH HALF OF DISPLACEMENT
U 021D, 0C85,759E,4000,25E7,0077,0 ;19684      NEXT/CO.BSBW-ALLIGN           ;
                                ;19685
                                ;19686         ;1110-------------------------;
                                ;19687         MDR_ZEXT(OSR),                ; GET DISPLACEMENT FROM OSR
                                ;19688         R[TEMP0]_ZEXT(XB) PC_PC+1,    ; GET HIGH HALF OF DISPLACEMENT
U 021E, 0C85,759E,4000,25E7,0077,0 ;19689      NEXT/CO.BSBW-ALLIGN           ;
```

```
                                    ;19690          ;1111-----------------------------;
                                    ;19691          MDR_ZEXT(OSR),                    ; GET DISPLACEMENT FROM OSR
                                    ;19692          R[TEMP0]_ZEXT(XB) PC_PC+1,        ; GET HIGH HALF OF DISPLACEMENT
U 021F, 0C85,759E,4000,25E7,0077,0  ;19693          NEXT/CO.BSBW-ALLIGN               ;
                                    ;19694
                                    ;19695 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                    ;19696 CO.BSBW-ALLIGN:
                                    ;19697          ;--------------------------------;
U 0770, 0881,2292,4020,0467,0077,1  ;19698          MDR_M[MDR].OR.(R[TEMP0].RR.24)    ; CONCATINATE TWO HALVES OF DISP
                                    ;19699
                                    ;19700          ;--------------------------------;
                                    ;19701          R[TEMP0]_M[PC],                   ; SAVE PC FOR LATER
U 0771, 0085,A592,4030,0047,0076,E  ;19702          NEXT/CO.BSB                       ;
```

```
                                ;19703 .TOC   ''       Control Instructions            : JSB, RSB''
                                ;19704
                                ;19705 ;*****************************************************************************
                                ;19706 ;       JSB     dest.ab
                                ;19707 ;          Input                          MDR             Address of subroutine
                                ;19708 ;
                                ;19709 ;       RSB
                                ;19710 ;          Input                          SP              Address of return address
                                ;19711 ;*****************************************************************************
                                ;19712 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;19713 CO.JSB:
                                ;19714         ;-------------------------------;
U 03A4, 0885,A592,4030,0047,0077,2  ;19715         R[TEMP0]_M[PC]                  ; SAVE PC FOR LATER PUSHING
                                ;19716
                                ;19717 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                ;19718         ;-------------------------------;
U 0772, 0881,2002,403D,8487,0077,3  ;19719         PC_M[MDR]                       ; STUFF PC WITH NEW ADDRESS
                                ;19720
                                ;19721         ;-------------------------------;
                                ;19722         VA_R[SP]_RB-CONX(4),            ; PUT ADDRESS POINTED TO BY SP INTO VA
U 0773, 0C85,473C,0027,84A7,0077,4  ;19723         PUSH RBS                        ; MAKE RESTART POSSIBLE IF WE DIE
                                ;19724
                                ;19725         ;-------------------------------;
                                ;19726         WRITE M[TEMP0],                 ; PUSH SAVED PC ONTO STACK
                                ;19727         SIZE[LONG],                     ; PC IS LONGWORD
U 0774, 0080,0592,4120,05D8,003F,9  ;19728         IRD1                            ; NEXT !!
                                ;19729 .REGION/IRD1.R1L,IRD1.R1H
                                ;19730 2A0:
                                ;19731 CO.RSB:
                                ;19732         ;000******FORCE ADDRESS*********;
U 02A0, 0480,05BE,4037,84A7,0077,5  ;19733         VA_R[SP]                        ; SP HAS ADDRESS OF RETURN ADDRESS
                                ;19734
                                ;19735 .REGION/CONTRL.R1L,CONTRL.R1H/CONTRL.R2L,CONTRL.R2H/CONTRL.R3L,CONTRL.R3H
                                ;19736         ;-------------------------------;
                                ;19737         READ,                           ; GET RETURN ADDRESS
                                ;19738         R[SP]_RB+CONX(4),               ; BUMP SP BY 4
U 0775, 0084,073D,0027,8050,0039,6  ;19739         NEXT/CO.JMP                     ; FINNISH LIKE A JUMP
```

```
                                        ;19740 .TOC   ''       Control Instructions          : Ird Rom Definition''
;19741  .NOBIN
;19742
;19743  .ICODE  ;       OPS             REG                             MEM             OPS         FPA REG                 FPA MEM
;19744        9D: FPD [NOP][IE.OPCOD.DEC          ]                                   [NOP][IE.OPCOD.DEC          ],                  ;ACBB
;19745            IRD1[LOD][OS.RED                ]                                   [LOD][OS.RED                ]
;19746  .OCODE
;19747        9D: CNT0[LOD][OS.RED                ][OS.RED                ]   [LOD][OS.RED                ][OS.RED                ],
;19748            CNT1[LOD][CO.ACB-REG            ][CO.ACBB-MEM           ]   [LOD][CO.ACB-REG            ][CO.ACBB-MEM           ]
;19749
;19750  .ICODE
;19751        0F1: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC          ],                  ;ACBL
;19752            IRD1[LOD][OS.RED                ]                                   [LOD][OS.RED                ]
;19753  .OCODE
;19754        0F1: CNT0[LOD][OS.RED               ][OS.RED                ]   [LOD][OS.RED                ][OS.RED                ],
;19755            CNT1[LOD][CO.ACB-REG            ][CO.ACBL-MEM           ]   [LOD][CO.ACB-REG            ][CO.ACBL-MEM           ]
;19756
;19757  .ICODE
;19758        3D: FPD [NOP][IE.OPCOD.DEC          ]                                   [NOP][IE.OPCOD.DEC          ],                  ;ACBW
;19759            IRD1[LOD][OS.RED                ]                                   [LOD][OS.RED                ]
;19760  .OCODE
;19761        3D: CNT0[LOD][OS.RED                ][OS.RED                ]   [LOD][OS.RED                ][OS.RED                ],
;19762            CNT1[LOD][CO.ACB-REG            ][CO.ACBW-MEM           ]   [LOD][CO.ACB-REG            ][CO.ACBW-MEM           ]
;19763
;19764  .ICODE
;19765        0F3: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC          ],                  ;AOBLEQ
;19766            IRD1[LOD][OS.RED                ]                                   [LOD][OS.RED                ]
;19767  .OCODE
;19768        0F3: CNT0[LOD][CO.AOBLEQ            ][OS.MOD                ]   [LOD][CO.AOBLEQ             ][OS.MOD                ],
;19769            CNT1[NOP][CO.AOBLEQ             ][CO.AOBLEQ             ]   [NOP][CO.AOBLEQ             ][CO.AOBLEQ             ]
;19770
;19771  .ICODE
;19772        0F2: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC          ],                  ;AOBLSS
;19773            IRD1[LOD][OS.RED                ]                                   [LOD][OS.RED                ]
;19774  .OCODE
;19775        0F2: CNT0[LOD][CO.AOBLSS            ][OS.MOD                ]   [LOD][CO.AOBLSS             ][OS.MOD                ],
;19776            CNT1[NOP][CO.AOBLSS             ][CO.AOBLSS             ]   [NOP][CO.AOBLSS             ][CO.AOBLSS             ]
;19777
;19778  .ICODE
;19779        0E1: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC          ],                  ;BBC
;19780            IRD1[LOD][OS.RED                ]                                   [LOD][OS.RED                ]
;19781  .OCODE
;19782        0E1: CNT0[LOD][CO.BB-REG            ][CO.BBC                ]   [LOD][CO.BB-REG             ][CO.BBC                ],
;19783            CNT1[NOP][CO.BB-RET             ][CO.BB-RET             ]   [NOP][CO.BB-RET             ][CO.BB-RET             ]
;19784
;19785  .ICODE
;19786        0E5: FPD [NOP][IE.OPCOD.DEC         ]                                   [NOP][IE.OPCOD.DEC          ],                  ;BBCC
;19787            IRD1[LOD][OS.RED                ]                                   [LOD][OS.RED                ]
;19788  .OCODE
;19789        0E5: CNT0[LOD][CO.BB-REG            ][CO.BBCC               ]   [LOD][CO.BB-REG             ][CO.BBCC               ],
;19790            CNT1[NOP][CO.BB-RET             ][CO.BB-RET             ]   [NOP][CO.BB-RET             ][CO.BB-RET             ]
```

```
;19791  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;19792       0E7: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BBCCI
;19793            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19794  .OCODE
;19795       0E7: CNT0[LOD][CO.BB-REG         ][CO.BBCCI          ]  [LOD][CO.BB-REG         ][CO.BBCCI          ],
;19796            CNT1[NOP][CO.BB-RET         ][CO.BB-RET         ]  [NOP][CO.BB-RET         ][CO.BB-RET         ]
;19797
;19798  .ICODE
;19799       0E3: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BBCS
;19800            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19801  .OCODE
;19802       0E3: CNT0[LOD][CO.BB-REG         ][CO.BBCS           ]  [LOD][CO.BB-REG         ][CO.BBCS           ],
;19803            CNT1[NOP][CO.BB-RET         ][CO.BB-RET         ]  [NOP][CO.BB-RET         ][CO.BB-RET         ]
;19804
;19805  .ICODE
;19806       0E0: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BBS
;19807            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19808  .OCODE
;19809       0E0: CNT0[LOD][CO.BB-REG         ][CO.BBS            ]  [LOD][CO.BB-REG         ][CO.BBS            ],
;19810            CNT1[NOP][CO.BB-RET         ][CO.BB-RET         ]  [NOP][CO.BB-RET         ][CO.BB-RET         ]
;19811
;19812  .ICODE
;19813       0E4: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BBSC
;19814            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19815  .OCODE
;19816       0E4: CNT0[LOD][CO.BB-REG         ][CO.BBSC           ]  [LOD][CO.BB-REG         ][CO.BBSC           ],
;19817            CNT1[NOP][CO.BB-RET         ][CO.BB-RET         ]  [NOP][CO.BB-RET         ][CO.BB-RET         ]
;19818
;19819  .ICODE
;19820       0E2: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BBSS
;19821            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19822  .OCODE
;19823       0E2: CNT0[LOD][CO.BB-REG         ][CO.BBSS           ]  [LOD][CO.BB-REG         ][CO.BBSS           ],
;19824            CNT1[NOP][CO.BB-RET         ][CO.BB-RET         ]  [NOP][CO.BB-RET         ][CO.BB-RET         ]
;19825
;19826  .ICODE
;19827       0E6: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BBSSI
;19828            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19829  .OCODE
;19830       0E6: CNT0[LOD][CO.BB-REG         ][CO.BBSSI          ]  [LOD][CO.BB-REG         ][CO.BBSSI          ],
;19831            CNT1[NOP][CO.BB-RET         ][CO.BB-RET         ]  [NOP][CO.BB-RET         ][CO.BB-RET         ]
;19832
;19833  .ICODE
;19834        1E: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BCC
;19835            IRD1[LOD][CO.BRCND          ]                          [LOD][CO.BRCND          ]
;19836  .OCODE
;19837        1E: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;19838            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;19839
;19840  .ICODE
;19841        1F: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BCS
;19842            IRD1[LOD][CO.BRCND          ]                          [LOD][CO.BRCND          ]
;19843  .OCODE
;19844        1F: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;19845            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
```

```
;19846  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;19847       013: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BEQL
;19848            IRD1[LOD][CO.BRCND          ]                          [LOD][CO.BRCND          ]
;19849  .OCODE
;19850       013: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;19851            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;19852
;19853  ;SEE BEQL                                                                                                  BEQLU
;19854
;19855  .ICODE
;19856       18: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BGEQ
;19857            IRD1[LOD][CO.BRCND          ]                          [LOD][CO.BRCND          ]
;19858  .OCODE
;19859       18: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;19860            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;19861
;19862  ;SEE BCC                                                                                                  BGEQU
;19863
;19864  .ICODE
;19865       14: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BGTR
;19866            IRD1[LOD][CO.BRCND          ]                          [LOD][CO.BRCND          ]
;19867  .OCODE
;19868       14: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;19869            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;19870
;19871  .ICODE
;19872       1A: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BGTRU
;19873            IRD1[LOD][CO.BRCND          ]                          [LOD][CO.BRCND          ]
;19874  .OCODE
;19875       1A: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;19876            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;19877
;19878  .ICODE
;19879       0E9: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BLBC
;19880            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19881  .OCODE
;19882       0E9: CNT0[NOP][CO.BLBC           ][CO.BLBC           ]  [NOP][CO.BLBC           ][CO.BLBC           ],
;19883            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;19884
;19885  .ICODE
;19886       0E8: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BLBS
;19887            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19888  .OCODE
;19889       0E8: CNT0[NOP][CO.BLBS           ][CO.BLBS           ]  [NOP][CO.BLBS           ][CO.BLBS           ],
;19890            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;19891
;19892  .ICODE
;19893       15: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                  ;BLEQ
;19894            IRD1[LOD][CO.BRCND          ]                          [LOD][CO.BRCND          ]
;19895  .OCODE
;19896       15: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;19897            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
```

```
;19898  .ICODE  ;       OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;19899       01B: FPD [NOP][IE.OPCOD.DEC      ]                              [NOP][IE.OPCOD.DEC      ],                    ;BLEQU
;19900            IRD1[LOD][CO.BRCND          ]                              [LOD][CO.BRCND          ]
;19901  .OCODE
;19902       01B: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;19903            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19904
;19905  .ICODE
;19906       019: FPD [NOP][IE.OPCOD.DEC      ]                              [NOP][IE.OPCOD.DEC      ],                    ;BLSS
;19907            IRD1[LOD][CO.BRCND          ]                              [LOD][CO.BRCND          ]
;19908  .OCODE
;19909       019: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;19910            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19911
;19912  ;SEE BCS                                                                                                      BLSSU
;19913
;19914  .ICODE
;19915       012: FPD [NOP][IE.OPCOD.DEC      ]                              [NOP][IE.OPCOD.DEC      ],                    ;BNEQ
;19916            IRD1[LOD][CO.BRCND          ]                              [LOD][CO.BRCND          ]
;19917  .OCODE
;19918       012: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;19919            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19920
;19921  ;SEE BNEQ                                                                                                     BNEQU
;19922
;19923  .ICODE
;19924       011: FPD [NOP][IE.OPCOD.DEC      ]                              [NOP][IE.OPCOD.DEC      ],                    ;BRB
;19925            IRD1[LOD][CO.BRB            ]                              [LOD][CO.BRB            ]
;19926  .OCODE
;19927       011: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;19928            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19929
;19930  .ICODE
;19931       031: FPD [NOP][IE.OPCOD.DEC      ]                              [NOP][IE.OPCOD.DEC      ],                    ;BRW
;19932            IRD1[LOD][CO.BRW            ]                              [LOD][CO.BRW            ]
;19933  .OCODE
;19934       031: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;19935            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19936
;19937  .ICODE
;19938       010: FPD [NOP][IE.OPCOD.DEC      ]                              [NOP][IE.OPCOD.DEC      ],                    ;BSBB
;19939            IRD1[LOD][CO.BSBB           ]                              [LOD][CO.BSBB           ]
;19940  .OCODE
;19941       010: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;19942            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19943
;19944  .ICODE
;19945       030: FPD [NOP][IE.OPCOD.DEC      ]                              [NOP][IE.OPCOD.DEC      ],                    ;BSBW
;19946            IRD1[LOD][CO.BSBW           ]                              [LOD][CO.BSBW           ]
;19947  .OCODE
;19948       030: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;19949            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
```

```
;19950  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;19951       01C: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;BVC
;19952            IRD1[LOD][CO.BRCND          ]                          [LOD][CO.BRCND          ]
;19953  .OCODE
;19954       01C: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;19955            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19956
;19957  .ICODE
;19958       01D: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;BVS
;19959            IRD1[LOD][CO.BRCND          ]                          [LOD][CO.BRCND          ]
;19960  .OCODE
;19961       01D: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;19962            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19963
;19964  .ICODE
;19965       08F: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;CASEB
;19966            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19967  .OCODE
;19968       08F: CNT0[LOD][OS.RED            ][OS.RED              ]  [LOD][OS.RED            ][OS.RED              ],
;19969            CNT1[LOD][CO.CASEB-REG      ][CO.CASEB-MEM        ]  [LOD][CO.CASEB-REG      ][CO.CASEB-MEM        ]
;19970
;19971  .ICODE
;19972       0CF: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;CASEL
;19973            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19974  .OCODE
;19975       0CF: CNT0[LOD][OS.RED            ][OS.RED              ]  [LOD][OS.RED            ][OS.RED              ],
;19976            CNT1[LOD][CO.CASEL-REG      ][CO.CASEL-MEM        ]  [LOD][CO.CASEL-REG      ][CO.CASEL-MEM        ]
;19977
;19978  .ICODE
;19979       0AF: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;CASEW
;19980            IRD1[LOD][OS.RED            ]                          [LOD][OS.RED            ]
;19981  .OCODE
;19982       0AF: CNT0[LOD][OS.RED            ][OS.RED              ]  [LOD][OS.RED            ][OS.RED              ],
;19983            CNT1[LOD][CO.CASEW-REG      ][CO.CASEW-MEM        ]  [LOD][CO.CASEW-REG      ][CO.CASEW-MEM        ]
;19984
;19985  .ICODE
;19986       017: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;JMP
;19987            IRD1[LOD][OS.ADD            ]                          [LOD][OS.ADD            ]
;19988  .OCODE
;19989       017: CNT0[NOP][CO.JMP            ][CO.JMP              ]  [NOP][CO.JMP            ][CO.JMP              ],
;19990            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19991
;19992  .ICODE
;19993       016: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;JSB
;19994            IRD1[LOD][OS.ADD            ]                          [LOD][OS.ADD            ]
;19995  .OCODE
;19996       016: CNT0[NOP][CO.JSB            ][CO.JSB              ]  [NOP][CO.JSB            ][CO.JSB              ],
;19997            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
;19998
;19999  .ICODE
;20000       005: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;RSB
;20001            IRD1[NOP][CO.RSB            ]                          [NOP][CO.RSB            ]
;20002  .OCODE
;20003       005: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ],
;20004            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD          ]
```

```
;20005  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;20006       OF4: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                  ;SOBGEQ
;20007            IRD1[LOD][OS.SOB              ]                        [LOD][OS.SOB              ]
;20008  .OCODE
;20009       OF4: CNT0[NOP][CO.SOBGEQ           ][CO.SOBGEQ           ] [NOP][CO.SOBGEQ           ][CO.SOBGEQ           ],
;20010            CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ] [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
;20011
;20012  .ICODE
;20013       OF5: FPD [NOP][IE.OPCOD.DEC        ]                        [NOP][IE.OPCOD.DEC        ],                  ;SOBGTR
;20014            IRD1[LOD][OS.SOB              ]                        [LOD][OS.SOB              ]
;20015  .OCODE
;20016       OF5: CNT0[NOP][CO.SOBGTR           ][CO.SOBGTR           ] [NOP][CO.SOBGTR           ][CO.SOBGTR           ],
;20017            CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ] [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
```

```
;20018  .TOC    "       Control Instructions            : Dsize Rom Definition"
;20019  .DCODE
;20020
;20021  09D:    SIZE [BYTE] [BYTE] [BYTE] [   0] [   0] [   0]  ;ACBB
;20022  0F1:    SIZE [LONG] [LONG] [LONG] [   0] [   0] [   0]  ;ACBL
;20023  03D:    SIZE [WORD] [WORD] [WORD] [   0] [   0] [   0]  ;ACBW
;20024  0F3:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;AOBLEQ
;20025  0F2:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;AOBLSS
;20026  0E1:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;BBC
;20027  0E5:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;BBCC
;20028  0E7:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;BBCCI
;20029  0E3:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;BBCS
;20030  0E0:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;BBS
;20031  0E4:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;BBSC
;20032  0E2:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;BBSS
;20033  0E6:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;BBSSI
;20034  01E:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BCC
;20035  01F:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BCS
;20036  013:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BEQL
;20037  ;SEE BEQL                                               ;BEQLU
;20038  018:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BGEQ
;20039  ;SEE BCC                                                ;BGEQU
;20040  014:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BGTR
;20041  01A:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BGTRU
;20042  0E9:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;BLBC
;20043  0E8:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;BLBS
;20044  015:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BLEQ
;20045  01B:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BLEQU
;20046  019:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BLSS
;20047  ;SEE BCS                                                ;BLSSU
;20048  012:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BNEQ
;20049  ;SEE BNEQ                                               ;BNEQU
;20050  011:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BRB
;20051  031:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;BRW
;20052  010:    SIZE [BYTE] [   0] [   0] [   0] [   0] [   0]  ;BSBB
;20053  030:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;BSBW
;20054  01C:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BVC
;20055  01D:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BVS
;20056  08F:    SIZE [BYTE] [BYTE] [BYTE] [   0] [   0] [   0]  ;CASEB
;20057  0CF:    SIZE [LONG] [LONG] [LONG] [   0] [   0] [   0]  ;CASEL
;20058  0AF:    SIZE [WORD] [WORD] [WORD] [   0] [   0] [   0]  ;CASEW
;20059  017:    SIZE [BYTE] [   0] [   0] [   0] [   0] [   0]  ;JMP
;20060  016:    SIZE [BYTE] [   0] [   0] [   0] [   0] [   0]  ;JSB
;20061  005:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;RSB
;20062  0F4:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;SOBGEQ
;20063  0F5:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;SOBGTR
;20064
;20065  .UCODE
                                                ;20066 .BIN
```

```
                         .TOC "PCALL.MIC"
                         .TOC "Revision 18.0"
                         ;       Jeff Peng, Gerard Koeckhoven, Brian Allison

.NOBIN
.TOC "  Revision History"

; REV   EXPLAINATION

; 18    When execute the CALLS and CALLG instruction with write protected
;       stack area, the ERRCOD has to be set a proper value to indicate
;       ACV fault.
;
; 17    The "CALLS and CALLG" instructions have to read the
;       mask as a word instead of a longword.
; 16    Fix CALLx to pass proper address on ACV's to memory management
; 15    Fix CALLx and RET to do exact probe if approximate probe gets TNV.
; 14    Initial release.
                                      .BIN
```

```
                                ;20086 .TOC     ''      Procedure Call                  : CALLS, CALLG''
                                ;20087
                                ;20088 ;*************************************************************************************
                                ;20089 ;       CALLG arglst.ab, dst.ab
                                ;20090 ;       CALLS numarg.rl, dst.ab
                                ;20091 ;          Input                          Q              Arglst,Numarg
                                ;20092 ;                                         MDR            Dst
                                ;20093 ;*************************************************************************************
                                ;20094 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;20095 PC.CALLG:
                                ;20096 PC.CALLS:
                                ;20097         ;-------------------------------;
                                ;20098         VA_M[MDR],                      ; PUT SUBROUTINE ADDRESS IN VA
U 03A5, 0041,2002,403D,84A7,00C1,B ;20099      SET FLAG0                       ; NEGATIVE LOGIC !!
                                ;20100
                                ;20101 .REGION/PCALL.R1L,PCALL.R1H/PCALL.R2L,PCALL.R2H/PCALL.R3L,PCALL.R3H
                                ;20102         ;-------------------------------;
                                ;20103         R[TEMP0]_Q Q_M[MDR],            ; SAVE SUBROUTINE ADDRESS IN Q
                                ;20104         STEPC 14.,                      ; FOR USE IN COUNT ONES ROUTINE
U 0C1B, 04B5,200C,7670,0047,00C1,E ;20105      IR<2-0>?                        ; PUT ARGLIST, NUMARG IN TEMP0
                                ;20106
                                ;20107 =110
                                ;20108         ;110----------------------------; CALLG
                                ;20109         READ,SIZE[WORD],                ; GET ENTRY MASK
                                ;20110         M[TEMP6]_D_R[SP],               ; SAVE SP IN D AND TEMP 6
                                ;20111         WB<1-0>.NE.0?,                  ; TEST FOR ALLIGNED STACK
U 0C1E, 0886,65BE,6397,8050,00C1,C ;20112      NEXT/PC.CALLG-ALLIGN            ;
                                ;20113
                                ;20114         ;111----------------------------; CALLS
                                ;20115         READ,SIZE[WORD],                ; GET ENTRY MASK
                                ;20116         M[TEMP6]_D_R[SP],               ; SAVE SP IN D AND TEMP 6
                                ;20117         WB<1-0>.NE.0?,                  ; TEST FOR ALLIGNED STACK
U 0C1F, 0886,65BE,6397,8050,00C2,0 ;20118      NEXT/PC.CALLS-ALLIGN            ;
                                ;20119
                                ;20120 =0
                                ;20121 PC.CALLG-ALLIGN:
                                ;20122         ;0------------------------------; CALLG  STACK LONGWORD ALLIGNED
                                ;20123         VA_D-ZLITO[68.],                ; MINIMUM ADDRESS POSSIBLE
U 0C1C, 0D80,0C30,0032,24A7,00C2,3 ;20124      NEXT/PC.CALL-APPX.PROBE         ;
                                ;20125
                                ;20126         ;1------------------------------; CALLG  STACK NOT LONGWORD ALLIGNED
                                ;20127         M[TEMP2]_D_D.ANDNOT.ZLITO[3],   ; LONGWORD ALLIGN STACK
                                ;20128         CLEAR FLAG0,                    ; SO WE'LL KNOW ABOUT THIS LATER
U 0C1D, 0D06,2C33,A030,1847,00C1,C ;20129      NEXT/PC.CALLG-ALLIGN            ; CONTINUE AS IF STACK WAS ALLIGNED
                                ;20130
                                ;20131 =0
                                ;20132 PC.CALLS-ALLIGN:
                                ;20133         ;0------------------------------; CALLS  STACK LONGWORD ALLIGNED
                                ;20134         VA_D-ZLITO[72.],                ; MINIMUM ADDRESS POSSIBLE
U 0C20, 0180,0C30,0032,44A7,00C2,3 ;20135      NEXT/PC.CALL-APPX.PROBE         ;
                                ;20136
                                ;20137         ;1------------------------------; CALLS  STACK NOT LONGWORD ALLIGNED
                                ;20138         M[TEMP2]_D_D.ANDNOT.ZLITO[3],   ; LONGWORD ALLIGN STACK
                                ;20139         CLEAR FLAG0,                    ; SO WE'LL KNOW ABOUT THIS LATER
U 0C21, 0D06,2C33,A030,1847,00C2,0 ;20140      NEXT/PC.CALLS-ALLIGN            ; CONTINUE AS IF STACK WAS ALLIGNED
```

```
                                ;20141 7E5:
                                ;20142 PC.CALL-TNV:                             ; ** jump to PCS **
                                ;20143          ;01*******FORCE ADDRESS*********; APPROX PROBE FAILED, DO EXACT
                                ;20144          R[TEMP5]_ZEXT(M[TEMP5]),        ; GET MASK TO COUNT THE REGISTERS
                                ;20145          SIZE[WORD],                     ;
                                ;20146          DEC STEPC,PATCH,                ; STEP COUNTER <- 13
U 07E5, 189C,559E,45D1,4047,C0C2,5 ;20147       FLAG1?,NEXT/PC.CALL-ACV         ;
                                ;20148
                                ;20149 7E7:     ;11*******FORCE ADDRESS*********;
U 07E7, 0480,0036,4030,0047,00FB,1 ;20150       NEXT/IE.TNV.RBACK               ; EXACT PROBE FAILED.  FAULT
                                ;20151
                                ;20152 PC.CALL-APPX.PROBE:
                                ;20153          ;-------------------------------;
                                ;20154          PROBE WRITE?,SIZE[LONG],        ; TEST WRITE ACCESS
                                ;20155          M[TEMP3]_PSL,                   ; SAVE PSL
U 0C23, 0186,3EF6,47A0,809D,08C4,C 479* ;20156  PL_[16.]                         ; CONSTANT FOR LATER
                                ;20157 =0000
                                ;20158 =1100
                                ;20159          ;1100-----------------------------; TNV
                                ;20160          R[MM.TEMP1]_M[VA],              ; SAVE VA
U 0C4C, 0485,B592,4033,C047,0561,8 ;20161       PUSH,NEXT/MM.PRB.WRT.TBM        ; LET CHARLIE DO THE WORK
                                ;20162
                                ;20163 =1101    ;1101-----------------------------;
                                ;20164          R[TEMP5]_M[MDR],                ; MOVE MASK INTO TEMP5
U 0C4D, 0485,2592,45F1,4047,007E,5 ;20165       FLAG1?,NEXT/PC.CALL-TNV         ; HAVE WE DONE EXACT PROBE YET?
                                ;20166
                                ;20167 =1110    ;1110*****************************; ACV ** jump to PCS **
                                ;20168          R[TEMP5]_ZEXT(M[MDR]),          ; GET MASK TO COUNT THE REGISTERS
                                ;20169          SIZE[WORD],                     ;
                                ;20170          DEC STEPC,PATCH,                ; STEP COUNTER <- 13
U 0C4E, 189D,259E,45D1,4047,00C2,5 ;20171       FLAG1?,NEXT/PC.CALL-ACV         ;
                                ;20172
                                ;20173 =1111
                                ;20174          ;1111-----------------------------; ACCESS OK
                                ;20175          WB_M[MDR].AND.ZLIT12[3],        ; WBUS <- MASK .AND. 3000
                                ;20176          SET MM.NOINT,                   ; DON'T ALLOW ANYMORE INTERRUPTS
U 0C4F, 0161,2D52,0A30,1847,00C3,C ;20177       WX.EQ.0?                        ; TEST MBZ BITS OF MASK FOR ILLEGAL OPER
                                ;20178
                                ;20179 =0       ;0-----------------------------; MBZ BITS ARE SET
U 0C3C, 0C80,0036,4030,0047,00FF,8 ;20180       NEXT/IE.OPER.FAULT              ; GO TELL CHARLIE
                                ;20181
                                ;20182          ;1-----------------------------;
U 0C3D, 0885,2A77,0031,0047,00C2,9 ;20183       R[TEMP4]_M[MDR].ASL.P           ; TEMP4 <- MASK'0000
                                ;20184
                                ;20185          ;-------------------------------;
                                ;20186          M[TEMP5]_R[TEMP4],              ; SAVE SHIFTED MASK FOR TESTING LATER
U 0C29, 0C86,55BE,4671,0047,00C3,E ;20187       IR<2-0>?                        ; WHICH INSTRUCTION ???
                                ;20188
                                ;20189 =110     ;110-----------------------------; CALLG
                                ;20190          VA_R[SP]_D-CONX(4),             ; POINT STACK TO RIGHT PLACE
                                ;20191          PUSH RBS=,                      ; SAVE STATE IN CASE WE FAIL
U 0C3E, 0C85,4730,0027,84A7,00C3,1 ;20192       NEXT/PC.CALL-MASK.MASK          ;
                                ;20193
                                ;20194          ;111-----------------------------;
U 0C3F, 0080,073C,0027,84A7,00C2,B ;20195       VA_R[SP]-CONX(4)                ; DEC STACK BEFORE PUSHING
```

```
                                ;20196          ;--------------------------------;
U 0C2B, 0480,0592,4020,05D8,00C2,D  ;20197      WRITE M[TEMP0],SIZE[LONG]       ; PUSH NUMBER OF ARGS ONTO STACK
                                ;20198
                                ;20199          ;--------------------------------;
U 0C2D, 0586,0C30,0030,4047,00C2,F  ;20200      M[TEMP0]_D-ZLIT0[8]             ; TEMP0 GETS NEW SP
                                ;20201
                                ;20202          ;--------------------------------;
U 0C2F, 0C84,0592,4037,84A7,00C3,1  ;20203      VA_R[SP]_M[TEMP0]               ; SP <- D-8
                                ;20204
                                ;20205 PC.CALL-MASK.MASK:
                                ;20206          ;--------------------------------;
                                ;20207          MDR_ZEXT(M[MDR]),               ; THROW AWAY ANY HIGH WORD CRAP
U 0C31, 0481,2016,401D,8467,00C3,3  ;20208      SIZE[WORD]                      ;
                                ;20209
                                ;20210          ;--------------------------------;
U 0C33, 0181,2F92,0030,7C67,00C3,5  ;20211      MDR_M[MDR].AND.OLIT8[15.]       ; MASK OFF DV AND IV BITS OF MASK
                                ;20212
                                ;20213 PC.CALL-FIND.REG:
                                ;20214          ;--------------------------------;
                                ;20215          PL_MSS M[MDR],                  ; FIND MOST SIG BIT SET IN MDR
U 0C35, 0C81,29C2,4DFD,8047,00C4,A  ;20216      WX<15-0>.NE.0?                  ; FIND OUT IF MDR = 0
                                ;20217
                                ;20218 =10       ;10-------------------------------; MDR = 0
                                ;20219          R[TEMP4]_RB.OR.ZEXT(M[TEMP3]),  ; TEMP4 <- MASK'PSW
                                ;20220          SIZE[WORD],                     ; EXTEND FROM BIT 15
U 0C4A, 0084,3016,44D1,0047,00C4,8  ;20221      FLAG3?,NEXT/PC.CALL-DONE.REG    ; GO BACK TO PUSHR OR CONTINUE CALL
                                ;20222
                                ;20223          ;11-------------------------------;
                                ;20224          WB_PL*2,                        ; WBUS GETS PLATCH*2
U 0C4B, 0C80,0B3D,C23D,8047,08C0,0 385*  ;20225  WB<5-0>?                        ; GO PUSH THE PROPER REGISTER
```

```
                                ;20226 =000000
                                ;20227 PC.CALL-PUSH.R0:
                                ;20228          ;000000------------------------;
                                ;20229          WRITE R[R0],SIZE[LONG],         ; PUSH R0 ONTO STACK
U 0C00, 0480,05BE,4024,05D8,04C8,5 ;20230       PUSH,NEXT/PC.CALL-DEC.SP1       ;
                                ;20231
                                ;20232          ;000001------------------------; THIS IS THE EXIT POINT OF PUSH REG
                                ;20233          R[TEMP4] RB.OR.ZEXT(M[TEMP3]),  ; TEMP4 <- MASK'PSW
                                ;20234          SIZE[WORD],                     ; EXTEND FROM BIT 15
U 0C01, 0084,3016,44D1,0047,00C4,8 ;20235       FLAG3?,NEXT/PC.CALL-DONE.REG    ;
                                ;20236
                                ;20237 PC.CALL-PUSH.R1:
                                ;20238          ;000010------------------------;
                                ;20239          WRITE R[R1],SIZE[LONG],         ; PUSH R1 ONTO STACK
U 0C02, 0880,05BE,4024,45D8,04C8,4 ;20240       PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20241
                                ;20242          ;000011------------------------;
                                ;20243          MDR_M[MDR].AND.ZLIT0[1],        ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C03, 0D81,2C12,0030,0C67,00C3,5 ;20244       NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
                                ;20245
                                ;20246 PC.CALL-PUSH.R2:
                                ;20247          ;000100------------------------;
                                ;20248          WRITE R[R2],SIZE[LONG],         ; PUSH R2 ONTO STACK
U 0C04, 0880,05BE,4024,85D8,04C8,4 ;20249       PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20250
                                ;20251          ;000101------------------------;
                                ;20252          MDR_M[MDR].AND.ZLIT0[3],        ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C05, 0981,2C12,0030,1C67,00C3,5 ;20253       NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
                                ;20254
                                ;20255 PC.CALL-PUSH.R3:
                                ;20256          ;000110------------------------;
                                ;20257          WRITE R[R3],SIZE[LONG],         ; PUSH R3 ONTO STACK
U 0C06, 0C80,05BE,4024,C5D8,04C8,4 ;20258       PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20259
                                ;20260          ;000111------------------------;
                                ;20261          MDR_M[MDR].AND.ZLIT0[7],        ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C07, 0181,2C12,0030,3C67,00C3,5 ;20262       NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
```

```
                                ;20263 PC.CALL-PUSH.R4:
                                ;20264          ;001000-------------------------;
                                ;20265          WRITE R[R4],SIZE[LONG],         ; PUSH R4 ONTO STACK
U 0C08, 0880,05BE,4025,05D8,04C8,4  ;20266      PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20267
                                ;20268          ;001001-------------------------;
                                ;20269          MDR_M[MDR].AND.ZLITO[0F],       ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C09, 0581,2C12,0030,7C67,00C3,5  ;20270      NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
                                ;20271
                                ;20272 PC.CALL-PUSH.R5:
                                ;20273          ;001010-------------------------;
                                ;20274          WRITE R[R5],SIZE[LONG],         ; PUSH R5 ONTO STACK
U 0C0A, 0C80,05BE,4025,45D8,04C8,4  ;20275      PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20276
                                ;20277          ;001011-------------------------;
                                ;20278          MDR_M[MDR].AND.ZLITO[1F],       ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C0B, 0181,2C12,0030,FC67,00C3,5  ;20279      NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
                                ;20280
                                ;20281 PC.CALL-PUSH.R6:
                                ;20282          ;001100-------------------------;
                                ;20283          WRITE R[R6],SIZE[LONG],         ; PUSH R6 ONTO STACK
U 0C0C, 0C80,05BE,4025,85D8,04C8,4  ;20284      PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20285
                                ;20286          ;001101-------------------------;
                                ;20287          MDR_M[MDR].AND.ZLITO[3F],       ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C0D, 0581,2C12,0031,FC67,00C3,5  ;20288      NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
                                ;20289
                                ;20290 PC.CALL-PUSH.R7:
                                ;20291          ;001110-------------------------;
                                ;20292          WRITE R[R7],SIZE[LONG],         ; PUSH R7 ONTO STACK
U 0C0E, 0880,05BE,4025,C5D8,04C8,4  ;20293      PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20294
                                ;20295          ;001111-------------------------;
                                ;20296          MDR_M[MDR].AND.ZLITO[7F],       ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C0F, 0181,2C12,0033,FC67,00C3,5  ;20297      NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
```

```
                                ;20298 PC.CALL-PUSH.R8:
                                ;20299          ;010000-------------------------;
                                ;20300          WRITE R[R8],SIZE[LONG],         ; PUSH R8 ONTO STACK
U 0C10, 0880,05BE,4026,05D8,04C8,4  ;20301      PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20302
                                ;20303          ;010001-------------------------;
                                ;20304          MDR_M[MDR].AND.ZLIT0[0FF],      ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C11, 0581,2C12,0037,FC67,00C3,5  ;20305      NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
                                ;20306
                                ;20307 PC.CALL-PUSH.R9:
                                ;20308          ;010010-------------------------;
                                ;20309          WRITE R[R9],SIZE[LONG],         ; PUSH R9 ONTO STACK
U 0C12, 0C80,05BE,402C,45D8,04C8,4  ;20310      PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20311
                                ;20312          ;010011-------------------------;
                                ;20313          MDR_M[MDR].AND.ZLIT0[1FF],      ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C13, 0181,2C12,003F,FC67,00C3,5  ;20314      NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
                                ;20315
                                ;20316 PC.CALL-PUSH.R10:
                                ;20317          ;010100-------------------------;
                                ;20318          WRITE R[R10],SIZE[LONG],        ; PUSH R10 ONTO STACK
U 0C14, 0C80,05BE,4026,85D8,04C8,4  ;20319      PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20320
                                ;20321          ;010101-------------------------;
                                ;20322          MDR_M[MDR].AND.OLIT8[3],        ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C15, 0D81,2F92,0030,1C67,00C3,5  ;20323      NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
                                ;20324
                                ;20325 PC.CALL-PUSH.R11:
                                ;2C 26          ;010110-------------------------;
                                ;20327          WRITE R[R11],SIZE[LONG],        ; PUSH R11 ONTO STACK
U 0C16, 0880,05BE,4026,C5D8,04C8,4  ;20328      PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20329
                                ;20330          ;010111-------------------------; FOR CHARLIE
                                ;20331          MDR_M[MDR].AND.OLIT8[7],        ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C17, 0581,2F92,0030,3C67,00C3,5  ;20332      NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
```

```
                                ;20333 PC.CALL-PUSH.R12:
                                ;20334          ;011000------------------------; FOR CHARLIE
                                ;20335          WRITE R[R12],SIZE[LONG],        ; PUSH R12 ONTO STACK
U 0C18, 0C80,05BE,4027,05D8,04C8,4 ;20336       PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20337
                                ;20338          ;011001------------------------; FOR CHARLIE
                                ;20339          MDR_M[MDR].AND.OLIT8[15.],      ; THROW AWAY BITS OF ALL PUSHED REGS
U 0C19, 0181,2F92,0030,7C67,00C3,5 ;20340       NEXT/PC.CALL-FIND.REG           ; GO FIND NEXT REGISTER TO PUSH
                                ;20341
                                ;20342 PC.CALL-PUSH.R13:
                                ;20343          ;011010------------------------; FOR CHARLIE
                                ;20344          WRITE R[R13],SIZE[LONG],        ; PUSH R13 ONTO STACK
U 0C1A, 0880,05BE,4027,45D8,04C8,4 ;20345       PUSH,NEXT/PC.CALL-DEC.SP        ; GO SUBTRACT 4 FROM SP
                                ;20346
                                ;20347 =100010
                                ;20348          ;100010------------------------;
                                ;20349          WB_M[MDR].AND.ZLIT0[1],         ; TEST THIS REGISTER'S BIT IN MASK
U 0C22, 0D81,2C12,0A30,0847,00C0,0 ;20350       WX.EQ.0?,NEXT/PC.CALL-PUSH.R0   ;
                                ;20351
                                ;20352 =100100
                                ;20353          ;100100------------------------;
                                ;20354          WB_M[MDR].AND.ZLIT0[2],         ; TEST THIS REGISTER'S BIT IN MASK
U 0C24, 0581,2C12,0A30,1047,00C0,2 ;20355       WX.EQ.0?,NEXT/PC.CALL-PUSH.R1   ;
                                ;20356
                                ;20357 =100110
                                ;20358          ;100110------------------------;
                                ;20359          WB_M[MDR].AND.ZLIT0[4],         ; TEST THIS REGISTER'S BIT IN MASK
U 0C26, 0981,2C12,0A30,2047,00C0,4 ;20360       WX.EQ.0?,NEXT/PC.CALL-PUSH.R2   ;
                                ;20361
                                ;20362 =101000
                                ;20363          ;101000------------------------;
                                ;20364          WB_M[MDR].AND.ZLIT0[8],         ; TEST THIS REGISTER'S BIT IN MASK
U 0C28, 0D81,2C12,0A30,4047,00C0,6 ;20365       WX.EQ.0?,NEXT/PC.CALL-PUSH.R3   ;
                                ;20366
                                ;20367 =101010
                                ;20368          ;101010------------------------;
                                ;20369          WB_M[MDR].AND.ZLIT0[10],        ; TEST THIS REGISTER'S BIT IN MASK
U 0C2A, 0581,2C12,0A30,8047,00C0,8 ;20370       WX.EQ.0?,NEXT/PC.CALL-PUSH.R4   ;
                                ;20371
                                ;20372 =101100
                                ;20373          ;101100------------------------;
                                ;20374          WB_M[MDR].AND.ZLIT0[20],        ; TEST THIS REGISTER'S BIT IN MASK
U 0C2C, 0D81,2C12,0A31,0047,00C0,A ;20375       WX.EQ.0?,NEXT/PC.CALL-PUSH.R5   ;
```

```
                                ;20376 =101110
                                ;20377          ;101110------------------------;
                                ;20378          WB_M[MDR].AND.ZLIT0[40],        ; TEST THIS REGISTER'S BIT IN MASK
U 0C2E, 0D81,2C12,0A32,0047,00C0,C ;20379       WX.EQ.0?,NEXT/PC.CALL-PUSH.R6   ;
                                ;20380
                                ;20381 =110000
                                ;20382          ;110000------------------------;
                                ;20383          WB_M[MDR].AND.ZLIT0[80],        ; TEST THIS REGISTER'S BIT IN MASK
U 0C30, 0581,2C12,0A34,0047,00C0,E ;20384       WX.EQ.0?,NEXT/PC.CALL-PUSH.R7   ;
                                ;20385
                                ;20386 =110010
                                ;20387          ;110010------------------------;
                                ;20388          WB_M[MDR].AND.ZLIT0[100],       ; TEST THIS REGISTER'S BIT IN MASK
U 0C32, 0581,2C12,0A38,0047,00C1,0 ;20389       WX.EQ.0?,NEXT/PC.CALL-PUSH.R8   ;
                                ;20390
                                ;20391 =110100
                                ;20392          ;110100------------------------;
                                ;20393          WB_M[MDR].AND.ZLIT4[32.],       ; TEST THIS REGISTER'S BIT IN MASK
U 0C34, 0981,2DD2,0A31,0047,00C1,2 ;20394       WX.EQ.0?,NEXT/PC.CALL-PUSH.R9   ;
                                ;20395
                                ;20396 =110110
                                ;20397          ;110110------------------------;
                                ;20398          WB_M[MDR].AND.ZLIT4[64.],       ; TEST THIS REGISTER'S BIT IN MASK
U 0C36, 0981,2DD2,0A32,0047,00C1,4 ;20399       WX.EQ.0?,NEXT/PC.CALL-PUSH.R10  ;
                                ;20400
                                ;20401 =111000
                                ;20402          ;111000------------------------; FOR CHARLIE
                                ;20403          WB_M[MDR].AND.ZLIT4[128.],      ; TEST THIS REGISTER'S BIT IN MASK
U 0C38, 0181,2DD2,0A34,0047,00C1,6 ;20404       WX.EQ.0?,NEXT/PC.CALL-PUSH.R11  ;
                                ;20405
                                ;20406 =111010
                                ;20407          ;111010------------------------; FOR CHARLIE
                                ;20408          WB_M[MDR].AND.ZLIT4[256.],      ; TEST THIS REGISTER'S BIT IN MASK
U 0C3A, 0981,2DD2,0A38,0047,00C1,8 ;20409       WX.EQ.0?,NEXT/PC.CALL-PUSH.R12  ;
                                ;20410 =
                                ;20411 =0
                                ;20412 PC.CALL-DONE.REG:
                                ;20413          ;0-----------------------------;
                                ;20414          R[TEMP7]_M[PC],                 ; SAVE PC
U 0C48, 0C85,A592,4031,C047,00C4,0 ;20415       NEXT/PC.CALL-PUSH.PC            ;
                                ;20416
                                ;20417          ;1-----------------------------;
                                ;20418          R[SP]_RB+CONX(4),                ; FIX STACK
U 0C49, 0484,073D,0127,8047,003F,9 ;20419       IRD1                            ; DO THIS FOR PUSHR
                                ;20420
```

```
                                        ;20421 =000
                                        ;20422 PC.CALL-PUSH.PC:
                                        ;20423          ;000-------------------------------;
                                        ;20424          WRITE M[TEMP7],SIZE[LONG],        ; PUSH OLD PC ONTO STACK
U 0C40, 0080,7592,4020,05D8,04C8,5      ;20425          PUSH,NEXT/PC.CALL-DEC.SP1         ;
                                        ;20426
                                        ;20427          ;001-------------------------------;
                                        ;20428          M[TEMP4]_MB.ANDNOT.ZLIT28[511.],; TEMP4 <- 0000'MASK'PSW'0000
U 0C41, 0586,4C53,867F,F847,00C4,2      ;20429          IR<2-0>?                          ;
                                        ;20430
                                        ;20431 PC.CALL-10:
                                        ;20432          ;010-------------------------------;
                                        ;20433          WRITE R[R13],SIZE[LONG],          ; SAVE FP ON STACK
U 0C42, 0080,05BE,4027,45D8,04C8,6      ;20434          PUSH,NEXT/PC.CALL-DEC.SP2         ; GO DEC STACK BY 4
                                        ;20435
                                        ;20436          ;011-------------------------------; EXECUTE FOR CALLS ONLY
                                        ;20437          M[TEMP4]_MB.OR.ZLIT28[2],         ; TEMP4 <- 0010'MASK'PSW'0000
U 0C43, 0986,4C52,4030,1047,00C4,2      ;20438          NEXT/PC.CALL-10                   ;
                                        ;20439
                                        ;20440          ;100-------------------------------;
U 0C44, 0C80,05BE,4027,05D8,00C4,5      ;20441          WRITE R[R12],SIZE[LONG]           ; SAVE AP ON STACK
                                        ;20442
                                        ;20443          ;101-------------------------------;
                                        ;20444          VA_R[SP]_RB-CONX(4),              ; DEC SP
U 0C45, 0084,073C,0427,84A7,00C5,0      ;20445          FLAG0?                            ;
                                        ;20446 =
                                        ;20447 =00
                                        ;20448          ;00--------------------------------; HERE IF STACK NOT LONGWORD ALLIGNED
                                        ;20449          PL_[30.],                         ; NUMBER TO SHIFT SP EXTRA BITS BY
U 0C50, 0D80,0EF6,4030,F047,04C8,7      ;20450          PUSH,NEXT/PC.CALL-SAVE.SP.BITS    ; GO PUT BITS IN MASK
                                        ;20451
                                        ;20452          ;01--------------------------------;
                                        ;20453          WRITE M[TEMP4].ANDNOT.ZLIT8[0FF],; CLEAR MBZ BITS OF PSL
                                        ;20454          SIZE[LONG],                       ; PUSH MASK WORD ONTO STACK
U 0C51, 0580,4D93,8027,FDD8,04C8,5      ;20455          PUSH,NEXT/PC.CALL-DEC.SP1         ;
                                        ;20456
                                        ;20457          ;10--------------------------------;
U 0C52, 0480,003A,403D,8487,00C3,7      ;20458          PC_Q                              ; POINT PC TO PROCEDURE
                                        ;20459 =
                                        ;20460
                                        ;20461          ;--------------------------------;
U 0C37, 0485,B592,4037,4047,00C3,9      ;20462          R[R13]_M[VA]                      ; FP <- SP
                                        ;20463
                                        ;20464          ;--------------------------------;
U 0C39, 0586,3E12,0038,8047,00C3,B      ;20465          M[TEMP3]_MB.AND.OLIT0[272.]       ; CLEAR BITS 3-0 OF PSL
                                        ;20466
                                        ;20467          ;--------------------------------;
                                        ;20468          WRITE ZLIT0[0],SIZE[LONG],        ; PROVIDE THE MAGIC CLEARED WORD
U 0C3B, 0580,0C37,0660,05D8,00C4,6      ;20469          IR<2-0>?                          ; WHICH CALL   FOR THE LAST TIME !!
```

```
                                ;20470 =110     ;110----------------------------; CALLG
                                ;20471          R[R12]_M[TEMP0],                ; AP <- ARGLIST
U 0C46  0084,0592,4037,0047,00C5,3 ;20472       NEXT/PC.CALL-TEST.IV.DV         ;
                                ;20473
                                ;20474          ;111--------------------------; CALLS
U 0C47, 0484,6710,0027,0047,00C5,3 ;20475       R[R12]_M[TEMP6]-CONX(4)         ; AP <- STACK ADDRESS OF NUMARG
                                ;20476
                                ;20477 PC.CALL-TEST.IV.DV:
                                ;20478          ;-------------------------------;
                                ;20479          WB_R[TEMP5],                    ; DRIVE LEFT JUST MASK TO WBUS
                                ;20480          PC_PC+2 MB_XB,                  ; ADD ANOTHER 2 TO PC FOR GOOD MEASURE
U 0C53, 0081,75BE,46D1,6047,00C5,4 ;20481       WB<31-30>?                      ; TEST IV DV BITS
                                ;20482
                                ;20483 =00
                                ;20484 PC.CALL-EXIT:
                                ;20485          ;00---------------------------;
                                ;20486          PSL_M[TEMP3],                   ; DON'T SET IV OR DV
U 0C54, 0C80,3002,413D,8007,003F,9 ;20487       IRDT                            ; GASP ...GASP  FINALLY !!!!!!!
                                ;20488
                                ;20489          ;01---------------------------;
                                ;20490          M[TEMP3]_MB.OR.ZLIT0[20],       ; SET IV
U 0C55, 0186,3C12,4031,0047,00C5,4 ;20491       NEXT/PC.CALL-EXIT               ;
                                ;20492
                                ;20493          ;10---------------------------;
                                ;20494          M[TEMP3]_MB.OR.ZLIT0[80],       ; SET DV
U 0C56, 0186,3C12,4034,0047,00C5,4 ;20495       NEXT/PC.CALL-EXIT               ;
                                ;20496
                                ;20497          ;11---------- ----------------;
                                ;20498          M[TEMP3]_MB.OR.ZLIT0[0A0],      ; SET IV AND DV
U 0C57, 0586,3C12,4035,0047,00C5,4 ;20499       NEXT/PC.CALL-EXIT               ;
                                ;20500
                                ;20501 PC.CALL-DEC.SP:
                                ;20502          ;-------------------------------;
                                ;20503          VA_R[SP]_RB-CONX(4),            ; DEC SP AND PLACE IN VA
U 0C84, 0884,073C,00A7,84A7,0002,0 ;20504       RETURN [32.]                    ; RETURN + 32
                                ;20505
                                ;20506 PC.CALL-DEC.SP1:
                                ;20507          ;-------------------------------;
                                ;20508          VA_R[SP]_RB-CONX(4),            ; DEC SP AND PLACE IN VA
U 0C85, 0884,073C,00A7,84A7,0000,1 ;20509       RETURN [T]                      ; RETURN + 1
                                ;20510
                                ;20511 PC.CALL-DEC.SP2:
                                ;20512          ;-------------------------------;
                                ;20513          VA_R[SP]_RB-CONX(4),            ; DEC SP AND PLACE IN VA
U 0C86, 0884,073C,00A7,84A7,0000,2 ;20514       RETURN [2]                      ; RETURN + 2
                                ;20515
                                ;20516 PC.CALL-SAVE.SP.BITS:
                                ;20517          ;-------------------------------;
                                ;20518          R[TEMP4]_RB.OR.(M[TEMP6].ASL.P),; TEMP4 <- SPX0'MASK'PSW'0000
U 0C87, 0884,6A7E,40B1,0047,0000,1 ;20519       RETURN [T]                      ;
```

```
                                ;20520 .TOC   "      Procedure Call                    : RET"
                                ;20521
                                ;20522 ;*****************************************************************************
                                ;20523 ;       RET
                                ;20524 ;          Input                          GPR 13          Pointer to stack frame
                                ;20525 ;*****************************************************************************
                                ;20526 .REGION/IRD1.R1L,IRD1.R1H
                                ;20527 =000
                                ;20528 PC.RET:
                                ;20529          ;000-------------------------------;
                                ;20530          VA_M[TEMP3]_R[R13]+CONX(4),     ; POINT VA TO MASK ON STACK
U 02B8, 04B6,373D,0027,44A7,00C8,8 ;20531       STEPC_14.                       ; FOR THE COUNT ONES ROUTINE
                                ;20532 =
                                ;20533 .REGION/PCALL.R1L,PCALL.R1H/PCALL.R2L,PCALL.R2H/PCALL.R3L,PCALL.R3H
                                ;20534          ;-------------------------------;
                                ;20535          READ,SIZE[LONG],                ; GET MASK WORD
U 0C88, 0180,3C11,0022,04B0,00C8,9 ;20536       VA_M[TEMP3]+ZLIT0[64.]          ; SET UP VA FOR MAX PROBE
                                ;20537
                                ;20538          ;-------------------------------;
                                ;20539          WB_M[MDR].AND.ZLIT24[32.],      ; RET FROM CALLS OR CALLG ???
U 0C89, 0981,2C92,0A31,0047,00C6,0 ;20540       WX.EQ.0?                        ;
                                ;20541
                                ;20542 =0       ;0-------------------------------;
                                ;20543          VA_VA+4,                        ; PROBE FOR NUMARG ALSO
                                ;20544          SL_[12.],                       ; CONSTANT FOR LATER
U 0C60, 0580,0F76,4030,6447,00C8,A ;20545       NEXT/PC.RET-PROBE               ;
                                ;20546
                                ;20547          ;1-------------------------------;
U 0C61, 0D80,0F76,4030,6047,00C8,A ;20548       SL_[12.]                        ; CONSTANT FOR LATER
                                ;20549
                                ;20550 PC.RET-PROBE:
                                ;20551          ;-------------------------------;
                                ;20552          PROBE READ?,SIZE[LONG],         ; ACCESS OK ???
U 0C8A, 0180,0EF6,47A0,805F,08C5,C 479* ;20553  PL_[16.]                        ;
                                ;20554 =0000
                                ;20555 =1100    ;1100-----------------------------; TNV
                                ;20556          R[MM.TEMP1]_M[VA],              ; SAVE VA
U 0C5C, 0485,B592,4033,C047,0562,8 ;20557       PUSH,NEXT/MM.PRB.READ.TBM       ; LET CHARLIE DO THE WORK
                                ;20558
                                ;20559 =1101    ;1101-----------------------------;
                                ;20560          R[TEMP5]_M[MDR].XZ,             ; GET MASK INTO TEMP5
U 0C5D, 0485,2077,05F1,4047,007E,5 ;20561       FLAG1?,NEXT/PC.CALL-TNV         ; WAS THIS EXACT OR APPROX PROBE?
                                ;20562
                                ;20563 =1110    ;1110-----------------------------; ACV
                                ;20564          R[TEMP5]_M[MDR].XZ,             ; GET MASK TO COUNT THE REGISTERS.
                                ;20565          DEC STEPC,                      ; STEP COUNTER <- 13
U 0C5E, 0C9D,2077,05F1,4047,00C2,5 ;20566       FLAG1?,NEXT/PC.CALL-ACV         ;
                                ;20567
                                ;20568 =1111    ;1111-----------------------------; ACCESS OK !!
                                ;20569          VA_M[TEMP3]+CONX(4),            ; POINT VA TO AP ON STACK
U 0C5F, 0860,3711,0020,04A7,00C8,8 ;20570       SET MM.NOINT                    ; DON'T ALLOW ANYMORE INTERRUPTS
```

```
                                    ;20571          ;--------------------------------;
                                    ;20572          WB_M[MDR].AND.ZLIT8[255.],       ; TEST MBZ BITS OF PSW
U 0C8B, 0D81,2D92,0A37,F847,00C6,4  ;20573          WX.EQ.0?                         ;
                                    ;20574
                                    ;20575 =0       ;0--------------------------------;
U 0C64, 0C80,0036,4030,0047,00FF,8  ;20576          NEXT/IE.OPER.FAULT               ; LET CHARLIE WORRY ABOUT IT
                                    ;20577
                                    ;20578          ;1--------------------------------;
                                    ;20579          R[TEMP2]_M[MDR],                 ; SAVE MASK IN TEMP2
                                    ;20580          READ,SIZE[LONG],                 ; GO GET AP
U 0C65, 0485,2592,4020,8450,00C8,C  ;20581          VA_VA+4                          ; POINT TO NEXT LONGWORD ON STACK
                                    ;20582
                                    ;20583          ;--------------------------------;
                                    ;20584          R[R12]_M[MDR],                   ; RESTORE AP
                                    ;20585          READ,SIZE[LONG],                 ; GET OLD FP
U 0C8C, 0C85,2592,4027,0450,00C8,D  ;20586          VA_VA+4                          ; POINT TO NEXT LONGWORD ON STACK
                                    ;20587
                                    ;20588          ;--------------------------------;
                                    ;20589          R[R13]_M[MDR],                   ; RESTORE FP
                                    ;20590          READ,SIZE[LONG],                 ; GET RETURN PC
U 0C8D, 0885,2592,4027,4450,00C8,E  ;20591          VA_VA+4                          ; POINT TO NEXT LONGWORD ON STACK
                                    ;20592
                                    ;20593          ;--------------------------------;
U 0C8E, 0081,2002,403D,8487,00C8,F  ;20594          PC_M[MDR]                        ; SET PC TO RETURN ADDRESS
                                    ;20595
                                    ;20596          ;--------------------------------;
                                    ;20597          R[TEMP6]_M[TEMP2].XZ,            ; GET A CLEAN REGISTER SAVE MASK
U 0C8F, 0C84,2077,0A71,8047,08C6,6 335* ;20598      WX.NE.0?                         ;
                                    ;20599
                                    ;20600 =0
                                    ;20601 PC.RET-DONE.REG:
                                    ;20602          ;0--------------------------------; NO REGISTERS TO POP
                                    ;20603          WB_M[TEMP2],                     ; DRIVE MASK TO WBUS
U 0C66, 0C80,2592,46F0,0047,00C5,8  ;20604          WB<31-30>?,NEXT/PC.RET-FIX.STACK;
                                    ;20605
                                    ;20606          ;1--------------------------------; START TESTING REGISTERS
                                    ;20607          WB_M[TEMP6].AND.ZLIT0[1],        ; RESTORE REGISTER 0 ??
U 0C67, 0580,6C12,0A30,0847,00C6,8  ;20608          WX.EQ.0?                         ;
                                    ;20609
                                    ;20610 =0       ;0--------------------------------;
                                    ;20611          READ,SIZE[LONG],                 ; GET REG 0 FROM MEMORY
                                    ;20612          VA_VA+4,                         ; POINT TO NEXT LONGWORD ON STACK
                                    ;20613          RNUM_[0],                        ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C68, 0181,DC37,0020,0450,04C9,4  ;20614          PUSH,NEXT/PC.RET-WRITE.GPR       ; GO RESTORE REGISTER
                                    ;20615
                                    ;20616          ;1--------------------------------;
                                    ;20617          WB_M[TEMP6].AND.ZLIT0[2],        ; RESTORE REGISTER 1 ??
U 0C69, 0D80,6C12,0A30,1047,00C6,A  ;20618          WX.EQ.0?                         ;
```

```
                                        ;20619 =0       ;0--------------------------------;
                                        ;20620          READ,SIZE[LONG],                  ; GET REG 1 FROM MEMORY
                                        ;20621          VA_VA+4,                          ; POINT TO NEXT LONGWORD ON STACK
                                        ;20622          RNUM_[1],                         ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C6A, 0581,DC37,0020,0C50,04C9,4      ;20623          PUSH,NEXT/PC.RET-WRITE.GPR        ; GO RESTORE REGISTER
                                        ;20624
                                        ;20625          ;1--------------------------------;
                                        ;20626          WB_M[TEMP6].AND.ZLITO[4],         ; RESTORE REGISTER 2 ??
U 0C6B, 0180,6C12,0A30,2047,00C6,C      ;20627          WX.EQ.0?                          ;
                                        ;20628
                                        ;20629 =0       ;0--------------------------------;
                                        ;20630          READ,SIZE[LONG],                  ; GET REG 2 FROM MEMORY
                                        ;20631          VA_VA+4,                          ; POINT TO NEXT LONGWORD ON STACK
                                        ;20632          RNUM_[2],                         ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C6C, 0581,DC37,0020,1450,04C9,4      ;20633          PUSH,NEXT/PC.RET-WRITE.GPR        ; GO RESTORE REGISTER
                                        ;20634
                                        ;20635          ;1--------------------------------;
                                        ;20636          WB_M[TEMP6].AND.ZLITO[8],         ; RESTORE REGISTER 3 ??
U 0C6D, 0580,6C12,0A30,4047,00C6,E      ;20637          WX.EQ.0?                          ;
                                        ;20638
                                        ;20639 =0       ;0--------------------------------;
                                        ;20640          READ,SIZE[LONG],                  ; GET REG 3 FROM MEMORY
                                        ;20641          VA_VA+4,                          ; POINT TO NEXT LONGWORD ON STACK
                                        ;20642          RNUM_[3],                         ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C6E, 0181,DC37,0020,1C50,04C9,4      ;20643          PUSH,NEXT/PC.RET-WRITE.GPR        ; GO RESTORE REGISTER
                                        ;20644
                                        ;20645          ;1--------------------------------;
                                        ;20646          WB_M[TEMP6].AND.ZLITO[10],        ; RESTORE REGISTER 4 ??
U 0C6F, 0580,6C12,0A30,8047,00C7,0      ;20647          WX.EQ.0?                          ;
                                        ;20648
                                        ;20649 =0       ;0--------------------------------;
                                        ;20650          READ,SIZE[LONG],                  ; GET REG 4 FROM MEMORY
                                        ;20651          VA_VA+4,                          ; POINT TO NEXT LONGWORD ON STACK
                                        ;20652          RNUM_[4],                         ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C70, 0981,DC37,0020,2450,04C9,4      ;20653          PUSH,NEXT/PC.RET-WRITE.GPR        ; GO RESTORE REGISTER
                                        ;20654
                                        ;20655          ;1--------------------------------;
                                        ;20656          WB_M[TEMP6].AND.ZLITO[20],        ; RESTORE REGISTER 5 ??
U 0C71, 0D80,6C12,0A31,0047,00C7,2      ;20657          WX.EQ.0?                          ;
                                        ;20658
                                        ;20659 =0       ;0--------------------------------;
                                        ;20660          READ,SIZE[LONG],                  ; GET REG 5 FROM MEMORY
                                        ;20661          VA_VA+4,                          ; POINT TO NEXT LONGWORD ON STACK
                                        ;20662          RNUM_[5],                         ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C72, 0D81,DC37,0020,2C50,04C9,4      ;20663          PUSH,NEXT/PC.RET-WRITE.GPR        ; GO RESTORE REGISTER
                                        ;20664
                                        ;20665          ;1--------------------------------;
                                        ;20666          WB_M[TEMP6].AND.ZLITO[40],        ; RESTORE REGISTER 6 ??
U 0C73, 0D80,6C12,0A32,0047,00C7,4      ;20667          WX.EQ.0?                          ;
```

```
                                    ;20668 =0       ;0------------------------------;
                                    ;20669          READ,SIZE[LONG],                ; GET REG 6 FROM MEMORY
                                    ;20670          VA_VA+4,                        ; POINT TO NEXT LONGWORD ON STACK
                                    ;20671          RNUM_[6],                       ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C74, 0D81,DC37,0020,3450,04C9,4  ;20672          PUSH,NEXT/PC.RET-WRITE.GPR      ; GO RESTORE REGISTER
                                    ;20673
                                    ;20674          ;1------------------------------;
                                    ;20675          WB_M[TEMP6].AND.ZLIT0[80],      ; RESTORE REGISTER 7 ??
U 0C75, 0580,6C12,0A34,0047,00C7,6  ;20676          WX.EQ.0?                        ;
                                    ;20677
                                    ;20678 =0       ;0------------------------------;
                                    ;20679          READ,SIZE[LONG],                ; GET REG 7 FROM MEMORY
                                    ;20680          VA_VA+4,                        ; POINT TO NEXT LONGWORD ON STACK
                                    ;20681          RNUM_[7],                       ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C76, 0981,DC37,0020,3C50,04C9,4  ;20682          PUSH,NEXT/PC.RET-WRITE.GPR      ; GO RESTORE REGISTER
                                    ;20683
                                    ;20684          ;1------------------------------;
                                    ;20685          WB_M[TEMP6].AND.ZLIT0[100],     ; RESTORE REGISTER 8 ??
U 0C77, 0D80,6C12,0A38,0047,00C7,8  ;20686          WX.EQ.0?                        ;
                                    ;20687
                                    ;20688 =0       ;0------------------------------;
                                    ;20689          READ,SIZE[LONG],                ; GET REG 8 FROM MEMORY
                                    ;20690          VA_VA+4,                        ; POINT TO NEXT LONGWORD ON STACK
                                    ;20691          RNUM_[8],                       ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C78, 0581,DC37,0020,4450,04C9,4  ;20692          PUSH,NEXT/PC.RET-WRITE.GPR      ; GO RESTORE REGISTER
                                    ;20693
                                    ;20694          ;1------------------------------;
                                    ;20695          WB_M[TEMP6].AND.ZLIT4[32.],     ; RESTORE REGISTER 9 ??
U 0C79, 0180,6DD2,0A31,0047,00C7,A  ;20696          WX.EQ.0?                        ;
                                    ;20697
                                    ;20698 =0       ;0------------------------------;
                                    ;20699          READ,SIZE[LONG],                ; GET REG 9 FROM MEMORY
                                    ;20700          VA_VA+4,                        ; POINT TO NEXT LONGWORD ON STACK
                                    ;20701          RNUM_[9],                       ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C7A, 0181,DC37,0020,4C50,04C9,4  ;20702          PUSH,NEXT/PC.RET-WRITE.GPR      ; GO RESTORE REGISTER
                                    ;20703
                                    ;20704          ;1------------------------------;
                                    ;20705          WB_M[TEMP6].AND.ZLIT4[64.],     ; RESTORE REGISTER 10 ??
U 0C7B, 0180,6DD2,0A32,0047,00C7,C  ;20706          WX.EQ.0?                        ;
                                    ;20707
                                    ;20708 =0       ;0------------------------------;
                                    ;20709          READ,SIZE[LONG],                ; GET REG 10 FROM MEMORY
                                    ;20710          VA_VA+4,                        ; POINT TO NEXT LONGWORD ON STACK
                                    ;20711          RNUM_[10.],                     ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C7C, 0181,DC37,0020,5450,04C9,4  ;20712          PUSH,NEXT/PC.RET-WRITE.GPR      ; GO RESTORE REGISTER
                                    ;20713
                                    ;20714          ;1------------------------------;
                                    ;20715          WB_M[TEMP6].AND.ZLIT4[128.],    ; RESTORE REGISTER 11 ??
U 0C7D, 0980,6DD2,0A34,0047,00C7,E  ;20716          WX.EQ.0?                        ;
```

```
                                ;20717 =0       ;0--------------------------------;
                                ;20718          READ,SIZE[LONG],                  ; GET REG 11 FROM MEMORY
                                ;20719          VA_VA+4,                          ; POINT TO NEXT LONGWORD ON STACK
                                ;20720          RNUM_[11.],                       ; TELL SUBROUTINE WHICH REGISTER THIS IS
U 0C7E, 0581,DC37,0020,5C50,04C9,4  ;20721      PUSH,NEXT/PC.RET-WRITE.GPR        ;
                                ;20722
                                ;20723          ;1--------------------------------;
                                ;20724          WB_M[TEMP2],                      ; DRIVE MASK TO WBUS
U 0C7F, 0C80,2592,46F0,0047,00C5,8  ;20725      WB<31-30>?                        ; TEST BITS TO GLUE STACK BACK TOGETHER
                                ;20726
                                ;20727 =00
                                ;20728 PC.RET-FIX.STACK:
                                ;20729          ;00-------------------------------; STACK ALREADY OK
                                ;20730          R[SP]_M[VA],                      ; SP <- SP
U 0C58, 0C85,B592,4037,8047,00C9,1  ;20731      NEXT/PC.RET-WHICH.INST            ;
                                ;20732
                                ;20733          ;01-------------------------------;
                                ;20734          VA_R[SP]_M[VA]+CONX(1),           ; SP <- SP+1
U 0C59, 0885,B711,0007,84A7,00C9,1  ;20735      NEXT/PC.RET-WHICH.INST            ;
                                ;20736
                                ;20737          ;10-------------------------------;
                                ;20738          VA_R[SP]_M[VA]+CONX(2),           ; SP <- SP+2
U 0C5A, 0C85,B711,0017,84A7,00C9,1  ;20739      NEXT/PC.RET-WHICH.INST            ;
                                ;20740
                                ;20741          ;11-------------------------------;
U 0C5B, 0987,BC11,0030,1847,00C9,0  ;20742      MTEMP0_VA+ZLIT0[3]                ; TEMP <- NEXT SP
                                ;20743
                                ;20744          ;-------------------------------;
                                ;20745          VA_R[SP]_M[TEMP0],                ; FINALLY FIX SP
U 0C90, 0C84,0592,4037,84A7,00C9,1  ;20746      NEXT/PC.RET-WHICH.INST            ;
                                ;20747
                                ;20748 PC.RET-WHICH.INST:
                                ;20749          ;-------------------------------; CALLG
                                ;20750          WB_M[TEMP2].AND.ZLIT24[32.],      ; FIND OUT WHICH CALL WE'RE RETURNING
U 0C91, 0180,2C92,0A71,0047,00C8,0  ;20751      WX.NE.0?                          ;
                                ;20752
                                ;20753 =0
                                ;20754 PC.RET-EXIT:
                                ;20755          ;0--------------------------------; CALLG
                                ;20756          PSW_M[TEMP2],                     ; LOAD CC BITS
U 0C80, 0080,2002,413D,8027,003F,9  ;20757      IRDT                              ; DONE
                                ;20758
                                ;20759          ;1--------------------------------; CALLS
                                ;20760          READ,SIZE[LONG],                  ; GET NUMARG
U 0C81, 0080,0036,4020,0450,00C9,2  ;20761      VA_VA+4                           ; POINT STACK TO NEXT LONGWORD
                                ;20762
                                ;20763          ;-------------------------------;
                                ;20764          R[SP]_ZEXT(M[MDR]),               ; THROW AWAY ANY GARBAGE
U 0C92, 0485,259E,4007,8047,00C9,3  ;20765      SIZE[BYTE]                        ; ONLY 256. ARGS PLEASE
```

```
                                ;20766          ;--------------------------------;
                                ;20767          R[SP]_M[VA]+(RB.ASL.SIZ),       ; PUSH STACK PAST ARGUMENTS
                                ;20768          SIZE[LONG],                     ; ASL BY 2
U 0C93, 0885,B5D1,0027,8047,00C8,0      ;20769          NEXT/PC.RET-EXIT                ; LEAVE
                                ;20770
                                ;20771 PC.RET-WRITE.GPR:
                                ;20772          ;1-------------------------------;
                                ;20773          R[GPR.R]_M[MDR],                ; WRITE THE REGISTER
U 0C94, 0885,2592,40BC,C047,0000,1      ;20774          RETURN [T]                      ; RETURN +1
                                ;20775 =0*
                                ;20776 PC.CALL-ACV:
                                ;20777          ;0*------------------------------;
                                ;20778          M[TEMP5]_MB.ANDNOT.ZLIT12[15.], ; THROW AWAY DV, IV AND MBZ BITS
                                ;20779          DEC STEPC,                      ; STEP COUNTER <- 12
U 0C25, 059E,5D53,8030,7847,00C9,5      ;20780          NEXT/PC.CALL-COUNT.ONES         ;
                                ;20781
                                ;20782          ;1*------------------------------; NO ACCESS
U 0C27, 0480,0036,4030,0047,0168,6      ;20783          NEXT/MM_PCALL_ACV_ENTRY         ; CHARLIE'S PROBLEM NOW
                                ;20784
                                ;20785 2825:    ;0*==============================; ** u-code in PCS **
                                ;20786          M[TEMP5]_MB.ANDNOT.ZLIT12[15.], ; THROW AWAY DV, IV AND MBZ BITS
                                ;20787          DEC STEPC,                      ; STEP COUNTER <- 12
U 2825, 059E,5D53,8030,7847,00C9,5      ;20788          NEXT/PC.CALL-COUNT.ONES         ;
                                ;20789
                                ;20790 2827:    ;1*==============================; NO ACCESS ** u-code in PCS **
                                ;20791          M[ERRCOD]_MB.OR.ZLIT0[4],       ; SET PROPER ERROR CODE
U 2827, 0186,BC12,4030,2047,0168,6      ;20792          NEXT/MM_PCALL_ACV_ENTRY         ; CHARLIE'S PROBLEM NOW
                                ;20793
                                ;20794 PC.CALL-COUNT.ONES:
                                ;20795          ;--------------------------------;
                                ;20796          PL_MSS M[TEMP5],                ; FIND A SET BIT IN MASK
U 0C95, 0C80,59C2,4DFD,8047,00C6,2      ;20797          WX<15-0>.NE.0?                  ; REAL BIT 0 ???
                                ;20798
                                ;20799 =10      ;10------------------------------; MASK=0
                                ;20800          M[TEMP5]_STEPC,SET FLAG1,       ; GET NON-DENSITY INTO TEMP5
U 0C62, 044E,5036,4030,0187,00C9,6      ;20801          NEXT/PC.CALL-FIX.PROBE          ; SO WE'LL KNOW WE'VE BEEN HERE BEFORE
                                ;20802
                                ;20803          ;11------------------------------;
                                ;20804          M[TEMP5]_MB.ANDNOT.ZLITPL[1],   ; CLEAR THE NEW FOUND BIT
                                ;20805          DEC STEPC,                      ; KEEP TRACK OF HOW MANY
U 0C63, 0D9E,5AD3,8030,0847,00C9,5      ;20806          NEXT/PC.CALL-COUNT.ONES         ;
                                ;20807
                                ;20808 PC.CALL-FIX.PROBE:
                                ;20809          ;--------------------------------;
                                ;20810          M[TEMP5]_MB.AND.ZLIT0[1F],      ; CLEAR GARBAGE BITS OF STEPC
U 0C96, 0186,5C12,08B0,F847,00C8,2      ;20811          IR<2>?                          ; BRANCH ON CALL OF RET
                                ;20812
                                ;20813 =0       ;0-------------------------------; CALLS   GALLG
                                ;20814          VA_M[VA]+(R[TEMP5].ASL.SIZE),   ; GET # OF BYTES NEEDED
                                ;20815          SIZE[LONG],                     ; MULT # OF CLEAR BITS BY 4
U 0C82, 0C81,B5D1,0021,44A7,00C2,3      ;20816          NEXT/PC.CALL-APPX.PROBE         ;
                                ;20817
                                ;20818          ;1-------------------------------; RET
                                ;20819          VA_M[VA]-(R[TEMP5].ASL.SIZE),   ; GET # OF BYTES NEEDED
U 0C83, 0881,B5D0,0021,44A7,00C8,A      ;20820          SIZE[LONG],NEXT/PC.RET-PROBE    ; MULT # OF CLEAR BITS BY 4
```

```
                                        ;20821 .TOC   ""      Procedure Call                  : Ird Rom Definition""
;20822  .NOBIN
;20823
;20824  .ICODE  ;       OPS          REG                    MEM              OPS        FPA REG                FPA MEM
;20825       OFA: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC        ],                    ;CALLG
;20826            IRD1[LOD][OS.ADD              ]                          [LOD][OS.ADD              ]
;20827  .OCODE
;20828       OFA: CNT0[LOD][OS.ADD              ][OS.ADD          ]  [LOD][OS.ADD              ][OS.ADD          ],
;20829            CNT1[NOP][PC.CALLG            ][PC.CALLG        ]  [NOP][PC.CALLG            ][PC.CALLG        ]
;20830
;20831
;20832
;20833   'CODE
;20834       OFB: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC        ],                    ;CALLS
;20835            IRD1[LOD][OS.RED              ]                          [LOD][OS.RED              ]
;20836  .OCODE
;20837       OFB: CNT0[LOD][OS.ADD              ][OS.ADD          ]  [LOD][OS.ADD              ][OS.ADD          ],
;20838            CNT1[NOP][PC.CALLS            ][PC.CALLS        ]  [NOP][PC.CALLS            ][PC.CALLS        ]
;20839
;20840
;20841
;20842  .ICODE
;20843       004: FPD [NOP][IE.OPCOD.DEC        ]                          [NOP][IE.OPCOD.DEC        ],                    ;RET
;20844            IRD1[NOP][PC.RET              ]                          [NOP][PC.RET              ]
;20845  .OCODE
;20846       004: CNT0[NOP][IE.BAD.IRD          ][IE.BAD.IRD      ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD      ],
;20847            CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD      ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD      ]
;20848
;20849
;20850
;20851  .TOC   ""      Procedure Call                  : Dsize Rom Definition""
;20852  .DCODE
;20853
;20854  OFA:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;CALLG
;20855
;20856  OFB:    SIZE [LONG] [BYTE] [   0] [   0] [   0] [   0]  ;CALLS
;20857
;20858  004:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;RET
;20859
;20860  .UCODE
                                        ;20861 .BIN
```

```
                                        ;20862 .TOC "MISQUE.MIC"
                                        ;20863 .TOC "REVISION 33.0"
                                        ;20864 ;       Jeff Peng, J. Heom, G. Koeckhoven, Brian Allison, CHARLIE MCDOWELL
                                        ;20865
;20866  .NOBIN
;20867
;20868  .TOC "  Revision History"
;20869
;20870  ; REV   EXPLAINATION
;20871
;20872  ; 33    Fix the instruction of INSQHI with reserved operand.
;20873  ; 32    Save mmtemp3 prior to unlocking header in case of tb miss,
;20874  ;       caused by the entry purging the header from the tb,then restore.
;20875  ;       Check all 3 lsb of header, even if software inter-lock is set,
;20876  ;       and if bits 1 or 2 is set, take reserved operand fault
;20877  ;       Also acquire hardware interlock prior to unlocking header
;20878  ; 30    Fix problem in QU.UNLOCK routine to properly restore the registers
;20879  ;       when the instruction has faulted.
;20880  ; 29    Fix REMQxI reg mode dest to properly disable header .ne. addr check
;20881  ; 28    27-MAY-80
;20882  ;       Fix interlocked queues to properly page fault.
;20883  ; 27    Set stack flag on halt to tell console to attempt restart/boot.
;20884  ;       Fix interlocked queues to set MM.NOINT before trying to clear lock.
;20885  ; 26    Initial release.
                                        ;20886          .BIN
```

```
                        ;20887 .TOC "  Misc. and Queue                  : XFC"
                        ;20888
                        ;20889 ;*****************************************************************************
                        ;20890 ;       XFC             Extended Function Call
                        ;20891 ;
                        ;20892 ;       Resources       FLAG2           Used by exception micro code
                        ;20893 ;                       VA
                        ;20894 ;*****************************************************************************
                        ;20895
                        ;20896 .REGION/IRD1.R1L,IRD1.R1H
                        ;20897 =000
                        ;20898 MS.XFC: ;-------------------------------;
                        ;20899         VA_M[SCBB]+ZLIT0[14],           ; LOAD MACRO VECTOR ADDRESS
                        ;20900         SET FLAG2,                      ; NO MORE PARAMETERS TO PUSH.
U 02C8, 0950,EC11,0030,A4A7,00EC,D      ;20901         NEXT/IE.FAULT                   ;
                        ;20902 =
                        ;20903 .TOC "  Misc. and Queue                  : NOP"
                        ;20904
                        ;20905 ;*****************************************************************************
                        ;20906 ;       NOP             No Operation
                        ;20907 ;
                        ;20908 ;       Resources       PC              PC <- PC - 1
                        ;20909 ;*****************************************************************************
                        ;20910
                        ;20911 =000
                        ;20912 MS.NOP: ;-------------------------------;
                        ;20913         PC_M[PC]-ZLIT0[1],              ;
U 0300, 0D81,AC10,0030,0C87,0000,2      ;20914         NEXT/GL.NOP.IRD1                ;
                        ;20915 =
                        ;20916 .TOC "  Misc. and Queue                  : BPT"
                        ;20917
                        ;20918 ;*****************************************************************************
                        ;20919 ;       BPT             Breakpoint Fault
                        ;20920 ;
                        ;20921 ;       Resources       VA
                        ;20922 ;                       FLAG2           Used by exception code
                        ;20923 ;*****************************************************************************
                        ;20924
                        ;20925 .REGION/IRD1.R1L,IRD1.R1H
                        ;20926 =000
                        ;20927 MS.BPT:
                        ;20928         ;-------------------------------;
                        ;20929         VA_M[SCBB]+ZLIT0[2C],           ; LOAD MACRO VECTOR ADDRESS.
U 0308, 0D50,EC11,0031,64A7,00EC,D      ;20930         SET FLAG2,NEXT/IE.FAULT         ; NO MORE PARAMATERS TO PUSH.
                        ;20931 =
                        ;20932 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
```

```
                                ;20933 .TOC "  Misc. and Queue                  : HALT"
                                ;20934
                                ;20935 ;*************************************************************************
                                ;20936 ;       HALT            Halt
                                ;20937 ;
                                ;20938 ;       Resources       TEMP1           Hold PSL to check mode
                                ;20939 ;                       FLAG3           Used by exception code
                                ;20940 ;                       PC              PC <- PC-1
                                ;20941 ;                       CRAR            Used to set HALT bit
                                ;20942 ;
                                ;20943 ;       Output  HALT            Sets HALT bit in console register
                                ;20944 ;*************************************************************************
                                ;20945
                                ;20946 .REGION/IRD1.R1L,IRD1.R1H
                                ;20947 =000
                                ;20948 MS.HALT:
                                ;20949         ;-------------------------------;
U 0310, 0886,1036,4030,0087,00C9,7      ;20950         M[TEMP1]_PSL                    ;
                                ;20951 =
                                ;20952 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                                ;20953         ;-------------------------------;
                                ;20954         WB_M[TEMP1].AND.ZLIT24[3],      ;
U 0C97, 0980,1C92,0A30,1847,00CB,8      ;20955         WX.EQ.0?                        ;
                                ;20956
                                ;20957 =0      ;0------------------------------;
                                ;20958         CLEAR FLAG3,                    ; PUSH PCBACK-2.
U 0CB8, 0418,0036,4030,0047,003B,0      ;20959         NEXT/IE.OPCOD.DEC               ; TAKE A PRIVILEDGED INSTRUCTION FAULT.
                                ;20960
                                ;20961         ;1------------------------------;
U 0CB9, 0981,AC11,0030,0C87,00C9,8      ;20962         PC_M[PC]+ZLIT0[1]               ; POINT THE PC TO THE NEXT INSTR + 2.
                                ;20963
                                ;20964         ;-------------------------------;
                                ;20965         M[FPDOFFSET]_ZLIT0[6],          ; SET HALT ERROR CODE
U 0C98, 096E,CC37,0030,3047,0000,1      ;20966         SET STACK FLAG                  ; TELL CONSOLE TO LOOK AT FRONT PANEL
                                ;20967
                                ;20968 1:      ;******* INTERRUPTS AND EXCEPTIONS HANDLED IN WCS BRANCH TO 2001(HEX).
                                ;20969 CN.CONSOLE:                              ; ENTRY FROM BOOT CODE
                                ;20970 MS.HALT.MICRO:
                                ;20971         ;-------------------------------; ENTRY FOR MICRO CODE DETECTED HALTS.
U 0001, 0180,0D37,0036,0247,00C9,A      ;20972         CRAR_ZLIT16[0C0]                ; CRAR_3
                                ;20973
                                ;20974         ;-------------------------------;
                                ;20975         CONREGS_ZLIT16[90],             ; SET HALT, AND HALT PENDING.
U 0C9A, 0180,0D37,0034,82C7,0088,4      ;20976         NEXT/CN.TYPE.PC                 ;
```

```
                                ;20977 .TOC "  Misc. and Queue                  : INDEX INSTRUCTION"
                                ;20978
                                ;*************************************************************************
                                ;       INDEX   subscript.rl, low.rl, high.rl, size.rl, indexin.rl, indexout.wl
                                ;
                                ;       Input           Q               Subscript
                                ;                       MDR             Low limit
                                ;
                                ;       Resources       TEMP4           Pass multiplier to MULSUB.MDR_0
                                ;                       TEMP6           Pass multiplicand to MULSUB.MDR_0
                                ;                       TEMP7           Save subscript
                                ;                       RTEMP9          Save low limit
                                ;                       RTEMP10         Save high limit
                                ;                       D               Save subscript + indexin
                                ;                       Q
                                ;                       FLAG3           Set if range trap occurs.
                                ;
                                ;        Subroutines    OS.RED
                                ;                       OS.WRT1
                                ;                       MULSUB.MDR_0
                                ;                       IE.INDEX.RANGE
                                ;
                                ;*************************************************************************
                                ;21000
                                ;21001 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;21002 =000
                                ;21003 MS.INDEX:
                                ;21004          ;000-----------------------------;
                                ;21005          M[TEMP7] Q,                     ; SAVE SUBSCRIPT.
                                ;21006          LOD INC BRA?,                   ;
U 0318, 0086,703A,41BD,8047,0410,0  ;21007      PUSH,NEXT/OS.RED                ; FETCH HIGH LIMIT.
                                ;21008
                                ;21009          ;001-----------------------------;
                                ;21010          R[TEMP9] Q Q_D,                 ; SAVE LOW LIMIT.
                                ;21011          LOD INC BRA?,                   ;
U 0319, 0484,002C,71B2,4047,0410,0  ;21012      PUSH,NEXT/OS.RED                ; FETCH SIZE.
                                ;21013
                                ;21014          ;010-----------------------------;
                                ;21015          R[TEMP10] Q Q_D,                ; SAVE HIGH LIMIT.
                                ;21016          LOD INC BRA?,                   ;
U 031A, 0484,002C,71B2,8047,0410,0  ;21017      PUSH,NEXT/OS.RED                ; FETCH INDEXIN.
                                ;21018
                                ;21019          ;011-----------------------------;
                                ;21020          D_M[MDR]+R[TEMP7],              ; SAVE SUBSCRIPT + INDEXIN.
                                ;21021          LOD INC BRA?,                   ;
U 031B, 0C81,2001,21B1,C047,0414,0  ;21022      PUSH,NEXT/OS.WRT1               ; FETCH INDEXOUT.
                                ;21023
                                ;21024          ;100-----------------------------;
                                ;21025          R[TEMP6] D,                     ; SETUP MULIPLICAND FOR MULTIPLY.
U 031C, 0C84,05B2,4021,98C7,00CA,0  ;21026      SIZE[LONG],ALUS_SIGND           ;
                                ;21027 =
                                ;21028 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
```

```
                                ;21029 =000     ;000-------------------------------;
                                ;21030          M[TEMP4] Q,                       ; SETUP MULTIPLIER FOR MULTIPLY.
                                ;21031          SIGND CMP?,SIZE[LONG],            ;
U 0CA0, 0C86,403A,4B6D,8047,0403,0 ;21032       PUSH,NEXT/IL.MULSUB.MDR_0         ; COMPUTE (INDEXIN + SUBSCRIPT) * SIZE
                                ;21033
                                ;21034 =011     ;011-------------------------------;
                                ;21035          R[DST.R] M[MDR].OR.Q,             ; RESULT IS POSITIVE, USE Q.
                                ;21036          WRITE NOTREG,SIZE[LONG],CCOP2,    ;
U 0CA3, 0485,200A,402C,55DA,00CA,1 ;21037       NEXT/MS.INDEX.50                  ;
                                ;21038
                                ;21039          ;100-------------------------------;
                                ;21040          R[DST.R] M[MDR]-Q,                ; RESULT IS NEGATIVE, USE -Q.
U 0CA4, 0485,2008,002C,55DA,00CA,1 ;21041       WRITE NOTREG,SIZE[LONG],CCOP2     ;
                                ;21042 =
                                ;21043 MS.INDEX.50:
                                ;21044          ;-------------------------------;
                                ;21045          WB M[TEMP7]-R[TEMP9],             ;
U 0CA1, 0880,7000,0B62,4047,08C9,9 374* ;21046  SIZE[LONG],SIGND CMP?              ; IS SUBSCRIPT LESS THAN LOW LIMIT?
                                ;21047
                                ;21048 =01      ;01--------------------------------; SUBSCRIPT IS GREATER OR EQUAL THAN LOW
                                ;21049          WB R[TEMP10]-M[TEMP7],            ;
                                ;21050          SIGND CMP?,SIZE[LONG],            ;
U 0C99, 0880,7003,0B62,8047,08CA,5 374* ;21051  NEXT/MS.INDEX.60                   ; IS SUBSCRIPT GREATER THAN HIGH LIMIT?
                                ;21052
                                ;21053          ;11--------------------------------; SUBSCRIPT IS LESS THAN LOW LIMIT
                                ;21054          SET FLAG3,                        ; PUSH PC ON TRAPS.
U 0C9B, 0C58,0036,4030,0047,00FB,A ;21055       NEXT/IE.INDEX.RANGE               ;
                                ;21056
                                ;21057 =01
                                ;21058 MS.INDEX.60:
                                ;21059          ;01--------------------------------; SUBSCRIPT IS LESS OR EQUAL THAN HIGH
U 0CA5, 0080,0036,4130,0047,003F,9 ;21060       IRD1                              ;
                                ;21061
                                ;21062          ;11--------------------------------; SUBSCRIPT IS GREATER THAN HIGH LIMIT
                                ;21063          SET FLAG3,                        ; PUSH PC ON TRAPS.
U 0CA7, 0C58,0036,4030,0047,00FB,A ;21064       NEXT/IE.INDEX.RANGE               ;
```

```
                                ;21065 .TOC "  Misc. and Queue                    : PUSHR"
                                ;21066
                                ;21067 ;*****************************************************************************
                                ;21068 ;       PUSHR   mask.rw
                                ;21069 ;
                                ;21070 ;       Input           MDR             Mask operand
                                ;21071 ;
                                ;21072 ;       Resources       TEMP0
                                ;21073 ;                       TEMP1
                                ;21074 ;                       TEMP2           Save zero extended mask operand
                                ;21075 ;                       TEMP4
                                ;21076 ;                       TEMP7
                                ;21077 ;                       RTEMP9
                                ;21078 ;                       RTEMP10
                                ;21079 ;                       Q
                                ;21080 ;                       VA
                                ;21081 ;
                                ;21082 ;       Subroutines     PRB.WRITE0
                                ;21083 ;                       PRB.WRITE1
                                ;21084 ;                       DEC.SP
                                ;21085 ;                       COUNT.ONES
                                ;21086 ;                       PC.CALL-FIND.REG
                                ;21087 ;
                                ;21088 ;        PUSHR FIRST DOES AN APPROXIMATE PROBE TO DETERMINE IF ALL 15 GPR'S
                                ;21089 ;        (NOT PC) CAN BE PUSHED ON THE STACK.  IF THE APPROXIMATE PROBE FAILS
                                ;21090 ;        THE NUMBER OF BITS SET IN THE MASK ARE COUNTED AND AN EXACT PROBE
                                ;21091 ;        IS DONE.  WHEN PROBEING, THE ADDRESS PASSED TO PRB.WRITEX IS
                                ;21092 ;        SP - N*4 WHERE N IS THE NUMBER OF REGISTERS TO BE PUSHED.
                                ;21093 ;
                                ;21094 ;*****************************************************************************
                                ;21095
                                ;21096 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;21097 MS.PUSHR:
                                ;21098          ;-------------------------------;
U 03A6, 0C86,05BE,4037,8047,00CA,2  ;21099      M[TEMP0]_R[SP]                  ; MOVE STACK POINTER TO AN MTEMP.
                                ;21100 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                                ;21101          ;-------------------------------;
U 0CA2, 0980,0C10,0031,E4A7,00CA,8  ;21102      VA_M[TEMP0]-ZLIT0[60.]          ; LOAD VA TO PROBE STACK.
                                ;21103
                                ;21104 =00      ;00-----------------------------; DO AN APPROXIMATE PROBE ASSUMING ALL
                                ;21105          PROBE WRITE?,                   ; 15 GPR'S WILL BE PUSHED (NOT PC).
                                ;21106          R[TEMP2]_ZEXT(M[MDR]),          ; SAVE ZERO EXTENDED MASK
                                ;21107          SIZE[WORD],                     ;
U 0CA8, 0C85,259E,4790,805D,0CC9,C 479*  ;21108 PUSH,NEXT/MS.PUSHR.PRB           ;
                                ;21109
                                ;21110          ;01-----------------------------; RETURN FROM COUNT.ONES,
                                ;21111          VA_R[SP]-M[TEMP0],              ; LOAD VA FOR EXACT PROBE.
U 0CA9, 0880,0003,0037,84A7,0579,A  ;21112      PUSH,NEXT/MM.PRB.WRITE.NR       ; NO RETURN IF ACV OR TNV.
                                ;21113
                                ;21114          ;10-----------------------------; EXACT PROBE IS OK,
                                ;21115          M[TEMP1]_ZLIT0[0E],             ; LOAD COUNTER FOR ADDRESSING GPR'S.
                                ;21116          SET MM.NOINT,                   ; STOP MEM MNGMT FROM TAKING INTERRUPTS.
U 0CAA, 0966,1C37,0030,7047,00CC,C  ;21117      NEXT/MS.PUSHR.PRB.OK            ;
                                ;21118 =
```

```
                                ;21119 =1100
                                ;21120 MS.PUSHR.PRB:
                                ;21121          ;1100--------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                ;21122          R[MM.TEMP1] M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 0C9C, 0485,B592,4033,C047,0561,8 ;21123       PUSH,NEXT/MM.PRB.WRT.TBM        ; LOAD THE PTE BACK IN THE TB.
                                ;21124
                                ;21125          ;1101--------------------------; RETURN+1 FROM MM.PRB.WRT.TBM,
                                ;21126          R[TEMP0] ZEXT(M[MDR]),          ; GET COUNT
                                ;21127          SIZE[WORD],STEPC 30.,           ; FOR EXACT PROBE.
U 0C9D, 00BD,259E,4010,0047,00CC,7 ;21128       NEXT/MS.COUNT.ONES              ;
                                ;21129
                                ;21130          ;1110--------------------------; NO ACCESS OR RETURN+2 FROM
                                ;21131          R[TEMP0] ZEXT(M[MDR]),          ; MM.PRB.WRT.TBM, GET COUNT
                                ;21132          SIZE[WORD],STEPC 30.,           ; FOR EXACT PROBE.
U 0C9E, 00BD,259E,4010,0047,00CC,7 ;21133       NEXT/MS.COUNT.ONES              ;
                                ;21134
                                ;21135          ;1111--------------------------; APPROXIMATE PROBE WAS OK,
                                ;21136          M[TEMP1] ZLIT0[0E],             ; LOAD COUNTER FOR ADDRESSING GPR'S.
U 0C9F, 0966,1C37,0030,7047,00CC,C ;21137       SET MM.NOINT                    ; STOP MEM MNGMT FORM TAKING INTERRUPTS.
                                ;21138
                                ;21139 =0
                                ;21140 MS.PUSHR.PRB.OK:
                                ;21141          ;0-----------------------------;
                                ;21142          Q M[MDR].SL.1,SIZE[WORD],       ; SHIFT BIT MASK AND EXAMINE BIT 15.
                                ;21143          SIGND CMP?,                     ; BRANCH ON SIGN BIT.
U 0CCC, 0081,2E52,DB50,0047,0CCB,1 354* ;21144  PUSH,NEXT/MS.PUSHR.LOOP         ;
                                ;21145
                                ;21146          ;1-----------------------------; RETURN FROM PUSHR.LOOP TO PUSH SP.
U 0CCD, 0480,073D,0027,85D8,00CB,1 ;21147       WRITE R[SP]+CONX(4),SIZE[LONG]  ; PUSH UNDECREMENTED SP.
                                ;21148
                                ;21149 =01
                                ;21150 MS.PUSHR.LOOP:
                                ;21151          ;01----------------------------;
                                ;21152          VA_R[SP]_RB-CONX(4),            ; PREPARE TO PUSH ON THE STACK.
                                ;21153          PUSH RBS=,                      ; SAVE IN CASE OF FAULT
U 0CB1, 0485,473C,0027,84A7,00CA,6 ;21154       NEXT/MS.PUSHR.MASK              ;
                                ;21155
                                ;21156 MS.DEC.SP:
                                ;21157          ;11----------------------------;
                                ;21158          VA_R[SP]_RB-CONX(4),            ; PREPARE TO PUSH ON THE STACK.
                                ;21159          PUSH RBS=,                      ; SAVE IN CASE OF FAULT
U 0CB3, 0885,473C,00A7,84A7,0000,1 ;21160       RETURN [+1]                     ;
                                ;21161
                                ;21162 MS.PUSHR.MASK:
                                ;21163          ;-----------------------------;
                                ;21164          MDR_M[TEMP2].AND.OLIT8[03F],    ; CLEAR MASK<15:14>
                                ;21165          SET FLAG3,                      ; SET FLAG TO INDICATE PUSHR
U 0CA6, 0558,2F92,0031,FC67,00C3,5 ;21166       NEXT/PC.CALL-FIND.REG           ; JUMP TO CALL TO FINISH PUSHING.
```

```
                                ;21167 .TOC "  Misc. and Queue                  : POPR"
                                ;21168
                                ;21169 ;*****************************************************************************
                                ;21170 ;       POPR    mask.rw
                                ;21171 ;
                                ;21172 ;       Input           MDR             Mask operand
                                ;21173 ;
                                ;21174 ;       Resources       TEMP0
                                ;21175 ;                       TEMP4
                                ;21176 ;                       TEMP7
                                ;21177 ;                       RTEMP9
                                ;21178 ;                       RTEMP10
                                ;21179 ;                       Q
                                ;21180 ;                       VA
                                ;21181 ;
                                ;21182 ;       Subroutines     PRB.READ0
                                ;21183 ;                       PRB.READ1
                                ;21184 ;                       DEC.SP
                                ;21185 ;                       COUNT.ONES
                                ;21186 ;
                                ;21187 ;       POPR FIRST DOES AN APPROXIMATE PROBE TO SEE IF ALL 15 GPR'S CAN BE
                                ;21188 ;       POPPED OF THE STACK(NOT PC).  IF THIS FAILS, THE NUMBER OF SET BITS
                                ;21189 ;       IN THE MASK ARE COUNTED AND AN EXACT PROBE IS DONE.  THE ADDRESS
                                ;21190 ;       PASSED TO PRB.READX IS SP + 4*N - 2. THE GENERAL FORM FOR POPPING
                                ;21191 ;       WOULD BE SP + 4*(N-1) BUT SINCE PRB.READX USES DSIZE AND DSIZE IS
                                ;21192 ;       WORD AND WE ARE INTERESTED IN POPPING LONGWORDS WE MUST PUSH THE
                                ;21193 ;       ADDRESS UP TWO MORE BYTES GIVING SP + 4*(N-1) + 2 WHICH IS EQUAL TO
                                ;21194 ;       THE ORIGINAL SP + 4*N - 2.
                                ;21195 ;
                                ;21196 ;*****************************************************************************
                                ;21197
                                ;21198 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;21199 MS.POPR:
                                ;21200          ;-------------------------------;
U 03A7, 0C86,05BE,4037,8047,00CA,B ;21201       M[TEMP0]_R[SP]                  ; MOVE SP TO AN MTEMP.
                                ;21202
                                ;21203 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                                ;21204
                                ;21205          ;-------------------------------;
U 0CAB, 0580,0C11,0031,DCA7,00CD,0 ;21206       VA_M[TEMP0]+ZLIT0[59.]          ;
                                ;21207
                                ;21208 =0       ;0-----------------------------; DO AN APPROXIMATE PROBE ASSUMING ALL
                                ;21209          PROBE READ?,SIZE[IDEP],         ; 15 GPR'S WILL BE POPPED (NOT PC).
U 0CD0, 0C80,0036,47B0 005F,0CCB,C 479* ;21210  PUSH,NEXT/MS.POPR.PRB            ; PUSH FOR JUMP TO COUNT.ONES
                                ;21211
                                ;21212          ;1-----------------------------; RETURN FROM COUNT.ONES,
U 0CD1, 0186,0C10,0030,1047,00CB,0 ;21213       M[TEMP0]_MB-ZLIT0[2]            ;
                                ;21214
                                ;21215          ;-------------------------------;
U 0CB0, 0480,0001,0037,84A7,00CB,2 ;21216       VA_M[TEMP0]+R[SP]               ; LOAD VA FOR EXACT PROBE.
```

```
                                            ;21217          ;-------------------------------;
U 0CB2, 0C80,0036,47B0,005F,08CA,C 479*     ;21218          PROBE READ?,SIZE[IDEP]          ; PROBE VA.
                                            ;21219
                                            ;21220 =1100    ;1100---------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                            ;21221          R[MM.TEMP1]_M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 0CAC, 0485,B592,4033,C047,0562,8          ;21222          PUSH,NEXT/MM.PRB.READ.TBM       ; LOAD THE PTE BACK IN THE TB.
                                            ;21223
                                            ;21224          ;1101---------------------------; CATCH RETURN+1 FROM MM.PRB.READ.TBM
U 0CAD, 0C80,0036,4030,0047,00F6,1          ;21225          NEXT/IE.TNV                     ; TAKE A TNV FAULT.
                                            ;21226
                                            ;21227          ;1110---------------------------; PROBE NO ACCESS OR RETURN+2 FROM
                                            ;21228          R[MM.TEMP3]_M[VA],              ; MM.PRB.READ.TBM,  SAVE FAULTING
U 0CAE, 0C85,B592,4039,C047,00F5,D          ;21229          NEXT/IE.ACV                     ; ADDRESS AND TAKE AN ACV FAULT.
                                            ;21230
                                            ;21231          ;1111---------------------------; WRITE ACCESS OK OR RETURN+3 FROM
                                            ;21232          VA_R[SP],                       ; MM.PRB.READ.TBM, EXACT PROBE IS OK
                                            ;21233          SET MM.NOINT,                   ; LOAD VA FOR FIRST POP.
U 0CAF, 0C60,05BE,4037,84A7,00CB,4          ;21234          NEXT/MS.POPR.PRB.OK             ; STOP MEM MNGMT FROM TAKING INTERRUPTS.
                                            ;21235
                                            ;21236 =1100
                                            ;21237 MS.POPR.PRB:
                                            ;21238          ;1100---------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                            ;21239          R[MM.TEMP1]_M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 0CBC, 0485,B592,4033,C047,0562,8          ;21240          PUSH,NEXT/MM.PRB.READ.TBM       ; LOAD THE PTE BACK IN THE TB.
                                            ;21241
                                            ;21242          ;1101---------------------------; RETURN+1 FROM MM.PRB.READ.TBM,
                                            ;21243          R[TEMP0]_ZEXT(M[MDR]),          ; GET COUNT
                                            ;21244          SIZE[WORD],STEPC_30.,           ; FOR EXACT PROBE.
U 0CBD, 00BD,259E,4010,0047,00CC,7          ;21245          NEXT/MS.COUNT.ONES              ;
                                            ;21246
                                            ;21247          ;1110---------------------------; NO ACCESS OR RETURN+2 FROM
                                            ;21248          R[TEMP0]_ZEXT(M[MDR]),          ; MM.PRB.READ.TBM, GET COUNT
                                            ;21249          SIZE[WORD],STEPC_30.,           ; FOR EXACT PROBE.
U 0CBE, 00BD,259E,4010,0047,00CC,7          ;21250          NEXT/MS.COUNT.ONES              ;
                                            ;21251
                                            ;21252          ;1111---------------------------; APPROXIMATE PROBE WAS OK,
                                            ;21253          VA_R[SP],                       ; LOAD VA FOR FIRST POP.
                                            ;21254          SET MM.NOINT,                   ; STOP MEM MNGMT FROM TAKING INTERRUPTS.
U 0CBF, 0C60,05BE,4037,84A7,00CB,4          ;21255          NEXT/MS.POPR.PRB.OK             ;
                                            ;21256
                                            ;21257 MS.POPR.PRB.OK:
                                            ;21258          ;-------------------------------;
U 0CB4, 0886,1E77,0030,0047,00CB,6          ;21259          M[TEMP1]_-1                     ;
                                            ;21260
                                            ;21261          ;-------------------------------;
                                            ;21262          Q_SEXT.M[MDR] STEPC_[16.],      ;
U 0CB6, 0181,2C1C,7010,8107,00CB,5          ;21263          SIZE[WORD]                      ;
```

```
                                        ;21264 =01
                                        ;21265 MS.POPR.LOOP:
                                        ;21266          ;01-------------------------------;
                                        ;21267          RNUM_R[TEMP1]_RB+1,             ; INCREMENT GPR ADDRESS.
                                        ;21268          DBZ STEPC?,                     ; CHECK TO SEE IF DONE.
U 0CB5, 0085,DE7C,0330,4047,00CD,2      ;21269          NEXT/MS.POPR.INNER              ;
                                        ;21270
                                        ;21271          ;11-------------------------------;
U 0CB7, 0084,073D,0027,84B0,00CC,3      ;21272          READ,VA_R[SP]_RB+CONX(4)        ; POP OFF OF THE STACK.
                                        ;21273
                                        ;21274          ;-------------------------------;
                                        ;21275          R[GPR.R]_M[MDR],                ; LOAD THE GPR.
U 0CC3, 0C85,2592,403C,C047,00CB,5      ;21276          NEXT/MS.POPR.LOOP               ;
                                        ;21277
                                        ;21278 =0
                                        ;21279 MS.POPR.INNER:
                                        ;21280          ;0-------------------------------;
                                        ;21281          Q_(R[ZERO]+Q).RR.1,             ; ROTATE BIT MASK AND TEST BIT<31>.
                                        ;21282          WB<31-30>?,                     ;
U 0CD2, 0080,0339,96FD,8047,08CB,5 355  ;21283          NEXT/MS.POPR.LOOP               ;
                                        ;21284
                                        ;21285          ;1-------------------------------;
U 0CD3, 0080,0036,4130,0047,003F,9      ;21286          IRD1                            ;
```

```
                        ;21287 .TOC "  Misc. and Queue                    :       COUNT.ONES ROUTINE"
                        ;21288
                        ;21289 ;*****************************************************************************
                        ;21290 ;       Entry points    MS.COUNT.ONES
                        ;21291 ;
                        ;21292 ;       Input           TEMP0           The word to be counted
                        ;21293 ;                       STEPC           30 decimal
                        ;21294 ;
                        ;21295 ;       Resources       STEPC           Holds the count
                        ;21296 ;                       PL              Used for finding next set bit
                        ;21297 ;
                        ;21298 ;       Output          TEMP0           4 times the number of set bits found
                        ;21299 ;
                        ;21300 ;       THIS ROUTINE CALCULATES THE NUMBER OF BITS SET IN TEMP0<15:0>.
                        ;21301 ;       STEPC IS INITIALIZED TO 30 BY THE CALLING ROUTINE, IT IS THEN
                        ;21302 ;       DECREMENTED FOR EACH SET BIT FOUND IN TEMP0<15:0>.  SET BITS
                        ;21303 ;       ARE DETECTED BY SCANNING FOR THE MOST SIGNIFICANT BIT SET AND
                        ;21304 ;       THEN CLEARING THAT BIT, COUNTING IT AND REPEATING.
                        ;21305 ;
                        ;21306 ;*****************************************************************************
                        ;21307
                        ;21308 MS.COUNT.ONES:
                        ;21309          ;-------------------------------; LOAD PL AND BRANCH IF MTEMP0=0
                        ;21310          PL_MSS M[TEMP0],                ;
U 0CC7, 0C80,09C2,4DFD,8047,00CB,A      ;21311          WX<15-0>.NE.0?                  ; SRKSTA<0>=WX<15:0>.NE.0
                        ;21312
                        ;21313 =10      ;10-----------------------------; TEMP0=0, NO MORE SET BITS.
                        ;21314          M[TEMP0]_STEPC,                 ; # BITS = 30. - STEPC
U 0CBA, 0486,0036,4030,0187,00CC,B      ;21315          NEXT/MS.COUNT.ONES10            ;
                        ;21316
                        ;21317          ;11-----------------------------; TEMP0 .NE. 0 MORE BITS TO COUNT.
                        ;21318          M[TEMP0]_MB.ANDNOT.ZLITPL[1],   ; CLEAR THIS BIT AND COUNT IT.
                        ;21319          DEC STEPC,                      ;
U 0CBB, 059E,0AD3,8030,0847,00CC,7      ;21320          NEXT/MS.COUNT.ONES              ;
                        ;21321
                        ;21322 MS.COUNT.ONES10:
                        ;21323          ;-------------------------------;
U 0CCB, 0186,0C12,0030,F847,00CF,C      ;21324          M[TEMP0]_MB.AND.ZLIT0[1F]       ; CLEAR UNDEFINED BITS FROM STEPC.
                        ;21325
                        ;21326          ;-------------------------------;
U 0CFC, 0986,0C13,0030,F047,00CF,D      ;21327          M[TEMP0]_ZLIT0[30.]-MB          ; LOAD NUMBER OF SET BITS INTO TEMP0.
                        ;21328
                        ;21329          ;-------------------------------;
                        ;21330          M[TEMP0]_(MB+R[TEMP0]).RL.1,    ; RESULT IS 4 TIMES NUMBER OF SET BITS.
U 0CFD, 0886,0301,C0B0,0047,0000,1      ;21331          RETURN [+1]                     ;
```

```
                                ;21332 .TOC "  Misc. and Queue                  : MOVPSL"
                                ;21333
                                ;21334 ;*************************************************************************
                                ;21335 ;       MOVPSL  dst.wl
                                ;21336 ;
                                ;21337 ;       Input (Dest is memory)          VA              Address of dst
                                ;21338 ;       Input (Dest is register)        RNUM            Register number of dst
                                ;21339 ;
                                ;21340 ;       Resources                       TEMP1           Temporary  to hold PSL
                                ;21341 ;
                                ;21342 ;       MOVPSL moves the PSL to the destination psecified by VA or RNUM.
                                ;21343 ;
                                ;21344 ;*************************************************************************
                                ;21345
                                ;21346 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;21347
                                ;21348 MS.MOVPSL:
                                ;21349         ;-------------------------------;
U 03A9, 0886,1036,4030,0087,00CF,E ;21350       M[TEMP1]_PSL                    ;
                                ;21351
                                ;21352 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                                ;21353
                                ;21354         ;-------------------------------;
U 0CFE, 0B80,067E,F807,F847,00CF,F ;21355       LONLIT_[3020FF00]               ; LOAD MASK OF MBZ BITS.
                                ;21356
                                ;21357         ;-------------------------------;
U 0CFF, 0886,1003,803D,4047,00D0,0 ;21358       M[TEMP1]_MB.ANDNOT.R[LONLIT]    ; FORCE MBZ BITS TO ZERO.
                                ;21359
                                ;21360         ;-------------------------------;
                                ;21361         R[DST.R] M[TEMP1],WRITE NOTREG, ;
U 0D00, 0C84,1592,412C,45DA,003F,9 ;21362      SIZE[LONG],IRD1                 ;
```

```
                                ;21363 .TOC "  Misc. and Queue                  : BISPSW, BICPSW"
                                ;21364
                                ;21365 ;*****************************************************************************
                                ;21366 ;       BISPSW  mask.rw
                                ;21367 ;       BICPSW  mask.rw
                                ;21368 ;
                                ;21369 ;       Input           MDR             Mask operand
                                ;21370 ;
                                ;21371 ;       Resources       TEMP1           Temporary to hold PSL
                                ;21372 ;                       FLAG3           Used by exception code if reserved oper
                                ;21373 ;
                                ;21374 ;       BISPSW AND BICPSW MODIFY PSW<7:0> BY ORING OR ANDNOTING
                                ;21375 ;       WITH A MASK WORD. A RESERVED OPERAND FAULT IS TAKEN AND PSW NOT
                                ;21376 ;       CHANGED IF MASK<15:8> IS NOT ZERO.
                                ;21377 ;
                                ;21378 ;*****************************************************************************
                                ;21379
                                ;21380 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;21381 =0
                                ;21382 MS.BISPSW:
                                ;21383         ;0-------------------------------;
                                ;21384         WB_M[MDR].AND.ZLIT8[OFF],       ; CHECK  MASK<15:8> MBZ
                                ;21385         WX.EQ.0?,                       ;
U 0292, 0D81,2D92,0A37,F847,04CD,4  ;21386         PUSH,NEXT/MS.BIXPSW.MBZ         ;
                                ;21387
                                ;21388         ;1-------------------------------;
                                ;21389         PSW_M[MDR].OR.R[TEMP1],         ;
U 0293, 0C81,2002,4030,4027,0000,2  ;21390         NEXT/GL.NOP.IRD1                ;
                                ;21391 =0
                                ;21392 MS.BICPSW:
                                ;21393         ;0-------------------------------;
                                ;21394         WB_M[MDR].AND.ZLIT8[OFF],       ; CHECK  MASK<15:8> MBZ
                                ;21395         WX.EQ.0?,                       ;
U 029E, 0D81,2D92,0A37,F847,C4CD,4  ;21396         PUSH,NEXT/MS.BIXPSW.MBZ         ;
                                ;21397
                                ;21398         ;1-------------------------------;
                                ;21399         PSW_M[MDR].NOTAND.R[TEMP1],     ;
U 029F, 0C81,2003,C030,4027,0000,2  ;21400         NEXT/GL.NOP.IRD1                ;
                                ;21401
                                ;21402 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                                ;21403
                                ;21404 =0
                                ;21405 MS.BIXPSW.MBZ:
                                ;21406         ;0-------------------------------; MASK<15:8> .NE. 0, RESERVED OPERAND.
                                ;21407         CLEAR FLAG3,                    ; PUSH PCBACK-2.
U 0CD4, 0418,0036,4030,0047,00FF,8  ;21408         NEXT/IE.OPER.FAULT              ;
                                ;21409
                                ;21410         ;1-------------------------------;
                                ;21411         M[TEMP1]_PSL,                   ;
U 0CD5, 0C86,1036,40B0,0087,0000,1  ;21412         RETURN [+1]                     ;
```

```
                                ;21413 .TOC "  Misc. and Queue                  : INSQUE INSTRUCTION"
                                ;21414
                                ;21415 ;*****************************************************************************
                                ;21416 ;       INSQUE  entry.ab, pred.ab
                                ;21417 ;
                                ;21418 ;       Input           Q               Entry operand
                                ;21419 ;                       MDR             Predecessor operand
                                ;21420 ;
                                ;21421 ;       Resources       TEMP1           Save entry
                                ;21422 ;                       TEMP2           Save pred
                                ;21423 ;                       TEMP3           Save (pred)+4
                                ;21424 ;                       TEMP4           Save (pred)
                                ;21425 ;                       VA
                                ;21426 ;
                                ;21427 ;       Subroutines     MM.PRB.WRITE.LONG
                                ;21428 ;
                                ;21429 ;*****************************************************************************
                                ;21430
                                ;21431 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;21432 QU.INSQUE:
                                ;21433         ;-------------------------------;
U 03AA, 0086,103A,403D,84A7,00CC,0 ;21434       VA_M[TEMP1]_R[ZERO],OR.Q        ; SAVE ENTRY.
                                ;21435 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                                ;21436 =00
                                ;21437         ;00-------------------------------;
                                ;21438         VA_M[TEMP1]+ZLIT0[4],           ;
U 0CC0, 0580,1C11,0030,24A7,0579,8 ;21439       PUSH,NEXT/MM.PRB.WRITE.LONG     ; PROBE ENTRY+4.
                                ;21440
                                ;21441         ;01-------------------------------;
                                ;21442         VA_R[TEMP2]_M[MDR],             ; SAVE PRED IN TEMP2.
U 0CC1, 0085,2592,4030,84A7,0579,8 ;21443       PUSH,NEXT/MM.PRB.WRITE.LONG     ; PROBE PRED.
                                ;21444
                                ;21445         ;10-------------------------------;
U 0CC2, 0880,0036,4020,0050,00CD,6 ;21446       READ.SIZE[LONG]                 ; GET (PRED).
                                ;21447 =
```

```
                                    ;21448 =0       ;0------------------------------;
                                    ;21449          VA_R[TEMP3]_M[MDR]+CONX(4),     ; SAVE (PRED)+4 IN TEMP3
U 0CD6, 0485,2711,0020,04A7,0579,8  ;21450          PUSH,NEXT/MM.PRB.WRITc.LONG     ; PROBE (PRED)+4.
                                    ;21451
                                    ;21452          ;1------------------------------; LOAD ADDRESS OF NEW ENTRY. ALL
                                    ;21453          VA_M[TEMP1]+ZLIT0[0],           ; REFERENCES ARE OK EXCEPT ENTRY, WHICH
U 0CD7, 0560,1C11,0030,04A7,00D0,1  ;21454          SET MM.NOINT                    ; WILL BE WRITTEN FIRST.
                                    ;21455
                                    ;21456          ;------------------------------;
                                    ;21457          WRITE,SIZE[LONG],               ; SET FORWARD LINK IN ENTRY TO (PRED)
U 0D01, 0485,2592,4021,05D8,00D0,2  ;21458          R[TEMP4]_M[MDR]                 ; SAVE (PRED) FOR SETTING CONDITION CODE
                                    ;21459
                                    ;21460          ;------------------------------;
U 0D02, 0480,0036,4030,0447,00D0,3  ;21461          VA_VA+4                         ;
                                    ;21462
                                    ;21463          ;------------------------------;
U 0D03, 0880,2592,4020,05D8,00CC,4  ;21464          WRITE M[TEMP2],SIZE[LONG]       ; SET BACK LINK IN ENTRY TO PRED.
                                    ;21465
                                    ;21466 =00      ;00-----------------------------;
                                    ;21467          VA_M[TEMP3]+ZLIT0[0],           ; LOAD VA WITH (PRED)+4.
U 0CC4, 0180,3C11,0030,04A7,04D0,4  ;21468          PUSH,NEXT/QU.WRITE.TEMP1        ; SET BACK LINK IN OLD SUCCESOR TO ENTRY
                                    ;21469
                                    ;21470          ;01-----------------------------;
                                    ;21471          VA_M[TEMP2]+ZLIT0[0],           ; LOAD VA WITH PRED.
U 0CC5, 0580,2C11,0030,04A7,04D0,4  ;21472          PUSH,NEXT/QU.WRITE.TEMP1        ; SET FORWARD LINK IN PRED TO ENTRY.
                                    ;21473
                                    ;21474          ;10-----------------------------;
                                    ;21475          WB_M[TEMP4]-R[TEMP2],           ; COMPARE FORWARD AND BACK LINKS
                                    ;21476          CCOP1,SIZE[LONG],               ; IN ENTRY TO SET CONDITION CODES.
U 0CC6, 0480,4000,0120,8847,003F,9  ;21477          IRD1                            ;
                                    ;21478 =
                                    ;21479
                                    ;21480 QU.WRITE.TEMP1:
                                    ;21481          ;------------------------------;
                                    ;21482          WRITE M[TEMP1],SIZE[LONG],      ;
U 0D04, 0C80,1592,40A0,05D8,0000,1  ;21483          RETURN [+1]                     ;
```

```
                        ;21484 .TOC "  Misc. and Queue                  : REMQUE"
                        ;21485
                        ;21486 ;*************************************************************************
                        ;21487 ;       REMQUE  entry.ab, addr.wl
                        ;21488 ;
                        ;21489 ;       Input (Dest is memory)          VA              Addr
                        ;21490 ;                                       Q               Entry operand address
                        ;21491 ;
                        ;21492 ;       Input (Dest is register)        RNUM            Addr
                        ;21493 ;                                       MDR             Entry operand address
                        ;21494 ;
                        ;21495 ;       Resources       MDR
                        ;21496 ;                       VA
                        ;21497 ;                       TEMP1           (entry+4)
                        ;21498 ;                       TEMP2           Entry
                        ;21499 ;                       TEMP3           (entry)+4 then (entry)
                        ;21500 ;                       TEMP4           Address of destination
                        ;21501 ;                       MM.NOINT
                        ;21502 ;
                        ;21503 ;       Subroutines     MM.PRB.WRITE.LONG
                        ;21504 ;                       MM.PRB.WRITE.SIZ
                        ;21505 ;                       OS.WRT1
                        ;21506 ;
                        ;21507 ;*************************************************************************
                        ;21508
                        ;21509 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;21510 QU.REMQUE.REG:
                        ;21511         ;-------------------------------;
                        ;21512         VA_R[TEMP2]_M[MDR],             ; SAVE ENTRY IN TEMP2.
U 03AB, 0885,2592,4030,84A7,00D0,5      ;21513         NEXT/QU.REMQUE.10               ;
                        ;21514 =0
                        ;21515 QU.REMQUE.MEM:
                        ;21516         ;0------------------------------;
                        ;21517         R[TEMP4]_M[VA],                 ; SAVE ADDR IN TEMP4 (DESTINATION OPER).
U 02A2, 0085,B592,4031,0047,0579,8      ;21518         PUSH,NEXT/MM.PRB.WRITE.LONG     ; PROBE ADDR.
                        ;21519
                        ;21520         ;1------------------------------;
U 02A3, 0886,203A,403D,84A7,00D0,5      ;21521         VA_M[TEMP2]_Q                   ; SAVE ENTRY IN TEMP2.
                        ;21522
                        ;21523 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                        ;21524
                        ;21525 QU.REMQUE.10:
                        ;21526         ;-------------------------------;
U 00D5, 0880,0036,4020,0450,00CD,8      ;21527         READ,SIZE[LONG],VA_VA+4         ; GET (ENTRY).
                        ;21528 =0
                        ;21529         ;0------------------------------;
                        ;21530         READ,SIZE[LONG],                ; GET (ENTRY)+4,
                        ;21531         VA_R[TEMP3]_M[MDR]+CONX(4),     ; SAVE (ENTRY)+4 IN TEMP3.
U 00D8, 0485,2711,0020,C4B0,0579,8      ;21532         PUSH,NEXT/MM.PRB.WRITE.LONG     ; PROBE (ENTRY)+4.
                        ;21533
                        ;21534         ;1------------------------------;
                        ;21535         VA_R[TEMP1]_M[MDR],             ; LOAD (ENTRY+4) IN VA & SAVE IN TEMP1.
U 00D9, 0065,2592,4030,44A7,00D0,6      ;21536         SET MM.NOINT                    ; ALL REFERENCES OK, DISABLE INT.
```

```
                                ;21537          ;--------------------------------;
                                ;21538          WRITE,SIZE[LONG],                ; SET FORWARD LINK OF PRED TO (ENTRY).
U 0D06, 0884,3710,0020,C5D8,00CD,A ;21539       R[TEMP3]_M[TEMP3]-CONX(4)        ; SAVE (ENTRY) TO SET CONDITION CODES.
                                ;21540 =0
                                ;21541          ;0------------------------------;
                                ;21542          VA_M[TEMP3]+ZLITO[4],            ; LOAD (ENTRY)+4
U 0CDA, 0980,3C11,0030,24A7,04D0,4 ;21543       PUSH,NEXT/QU.WRITE.TEMP1         ; SET BACK LINK OF SUCCESOR TO (ENTRY+4)
                                ;21544
                                ;21545          ;1------------------------------;
U 0CDB, 0D80,4C11,0030,04A7,00D0,7 ;21546       VA_M[TEMP4]+ZLITO[0]             ; LOAD ADDR(DESTINATION OPERAND ADDRESS)
                                ;21547
                                ;21548          ;--------------------------------;
                                ;21549          R[DST.R]_M[TEMP2],               ;
U 0D07, 0884,2592,402C,45DA,00D0,8 ;21550       WRITE NOTREG,SIZE[LONG]          ; SET POINTER TO REMOVED ENTRY.
                                ;21551
                                ;21552          ;--------------------------------;
                                ;21553          WB_M[TEMP3]-R[TEMP1],            ;
U 0D08, 0C80,3000,0020,4847,00D0,9 ;21554       CCOP1,SIZE[LONG]                 ; SET PSL<N,Z,C>.
                                ;21555
                                ;21556          ;--------------------------------;
                                ;21557          WB_M[TEMP2]-R[TEMP1],            ;
                                ;21558          CCOP2,SIZE[LONG],                ; SET PSL<V>.
U 0D09, 0480,2000,0120,5047,003F,9 ;21559       IRD1                             ;
```

```
;21560 .TOC "  Misc. and Queue                  : INSQHI"
;21561
;21562 ;*************************************************************************
;21563 ;       INSQHI  entry.ab, header.aq
;21564 ;
;21565 ;       Input           MDR             header
;21566 ;                       Q               entry
;21567 ;
;21568 ;       Resources       TEMP1           (header)
;21569 ;                       TEMP2           header - entry
;21570 ;                       TEMP4           header (needed to release lock on fault)
;21571 ;                       TEMP5           entry
;21572 ;                       D               successor
;21573 ;
;21574 ;       Subroutines     MM.PRB.WRITE.SIZ
;21575 ;                       QU.REMQHI.20
;21576 ;                       QU.SET.LOCK
;21577 ;
;21578 ;*************************************************************************
;21579
;21580 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
;21581 =0
;21582 QU.INSQHI:
;21583         ;0-----------------------------;
;21584         VA_R[TEMP4]_M[MDR],             ; LOAD ADDRESS OF HEADER AND SAVE IT.
U 02AC, 0885,2592,4031,04A7,04D2,9
;21585         PUSH,NEXT/QU.SET.LOCK           ; SET THE SECONDARY INTERLOCK.
;21586
;21587         ;1-----------------------------;
;21588         VA_D_M[TEMP1]+R[TEMP4],         ; LOAD ADDRESS OF CURRENT HEAD.
U 02AD, 0808,1001,2031,04A7,00CD,C
;21589         CLEAR FLAG1                     ; MAY BE SET BY QU.SET.LOCK FOR REMQXI
;21590 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
```

```
                                    ;21591 =0       ;0------------------------------;
                                    ;21592          CLEAR FLAG0,                    ; SET BY QU.SET.LOCK FOR CC SETTING
                                    ;21593          PROBE WRITE?,SIZE[IDEP],        ;
U OCDC, 0C00,0036,47B0,005D,0D68,4 479* ;21594      PUSH,NEXT/MM.PRB.WRITE.SIZ      ; CHECK THAT SUCCESSOR CAN BE WRITTEN.
                                    ;21595
                                    ;21596          ;1------------------------------;
U OCDD, 0884,4008,0030,8047,00D0,A  ;21597          R[TEMP2]_M[TEMP4]-Q             ; SAVE HEADER - ENTRY
                                    ;21598
                                    ;21599          ;------------------------------;
U OD0A, 0486,503A,403D,84A7,00D0,B  ;21600          VA_M[TEMP5]_R[ZERO].OR.Q        ; LOAD ADDRESS OF ENTRY AND SAVE IT.
                                    ;21601
                                    ;21602          ;------------------------------;
                                    ;21603          WB_M[TEMP5].AND.ZLIT0[7],       ; CHECK ENTRY FOR QUAD ALIGNMENT
U OD0B, 0960,5C12,0A30,3847,00CD,E  ;21604          SET MM.NOINT,WX.EQ.0?           ;
                                    ;21605
                                    ;21606 =0       ;0------------------------------; NOT ALIGNED, RESERVED OPERAND.
                                    ;21607          VA_M[TEMP4],                    ; LOAD ADDRESS OF HEADER
U OCDE, 0C80,4002,403D,84A7,00FF,8  ;21608          NEXT/IE.OPER.FAULT              ;
                                    ;21609
                                    ;21610          ;1------------------------------; ALIGNMENT IS OK.
                                    ;21611          WRITE,SIZE[LONG],               ; SET FORWARD LINK IN ENTRY
U OCDF, 0C80,1001,1020,85D8,00D0,C  ;21612          Q_M[TEMP1]+R[TEMP2]             ;
                                    ;21613
                                    ;21614          ;------------------------------;
U OD0C, 0180,5C11,0030,24A7,00D0,D  ;21615          VA_M[TEMP5]+ZLIT0[4]            ; LOAD ADDRESS OF ENTRY + 4
                                    ;21616
                                    ;21617          ;------------------------------;
U OD0D, 0080,2592,4020,05D8,00D0,E  ;21618          WRITE M[TEMP2],SIZE[LONG]       ; SET BACKWARD LINK IN ENTRY
                                    ;21619
                                    ;21620          ;------------------------------;
U OD0E, 0D80,0C31,0030,24A7,00D0,F  ;21621          VA_D+ZLIT0[4]                   ; LOAD ADDRESS OF SUCCESSOR + 4
                                    ;21622
                                    ;21623          ;------------------------------;
                                    ;21624          WRITE -Q,SIZE[LONG],            ; SET BACKWARD LINK IN SUCCESSOR
U 0D0F, 0880,0038,002D,85D8,00CE,A  ;21625          NEXT/QU.REMQHI.20               ; GO RELEASE SECONDARY INTERLOCK.
```

```
                                ;21626 .TOC "  Misc. and Queue                    : INSQTI"
                                ;21627
                                ;21628 ;*****************************************************************************
                                ;21629 ;       INSQTI  entry.ab, header.aq
                                ;21630 ;
                                ;21631 ;       Input           MDR             header
                                ;21632 ;                       Q               entry
                                ;21633 ;
                                ;21634 ;       Resources       TEMP1           (header)
                                ;21635 ;                       TEMP4           header (needed to release lock on fault)
                                ;21636 ;                       TEMP5           header - entry
                                ;21637 ;                       D               successor
                                ;21638 ;                       MDR             tail - header
                                ;21639 ;
                                ;21640 ;       Subroutines     MM.PRB.WRITE.SIZ
                                ;21641 ;                       QU. UNLOCK
                                ;21642 ;                       QU.SET.LOCK
                                ;21643 ;
                                ;21644 ;*****************************************************************************
                                ;21645
                                ;21646 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;21647 =0
                                ;21648 QU.INSQTI:
                                ;21649         ;0-------------------------------;
                                ;21650         VA_R[TEMP4]_M[MDR],             ; LOAD ADDRESS OF HEADER AND SAVE IT
U 02B2, 0885,2592,4031,04A7,04D2,9 ;21651      PUSH,NEXT/QU.SET.LOCK           ;
                                ;21652
                                ;21653         ;1-------------------------------;
                                ;21654         CLEAR FLAG1,                    ; MAY BE SET BY QU.SET.LOCK FOR REMQXI
U 02B3, 0508,4C11,0030,24A7,00D1,0 ;21655      VA_M[TEMP4]+ZLIT0[4]            ; LOAD ADDRESS OF HEADER + 4
                                ;21656 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                                ;21657         ;-------------------------------;
                                ;21658         CLEAR FLAG0,                    ; SET BY QU.SET.LOCK FOR CC SETTING
U 0D10, 0000,0036,4020,0050,00D1,1 ;21659      READ,SIZE[LONG]                 ; FETCH (HEADER + 4)
                                ;21660
                                ;21661         ;-------------------------------; LOAD ADDRESS OF TAIL
U 0D11, 0881,2001,0031,04A7,00D1,2 ;21662      VA_M[MDR]+R[TEMP4]              ; TAIL = HEADER + (HEADER + 4)
```

```
                                    ;21663          ;================================; ** patch to PCS **
                                    ;21664          WB_M[VA].AND.ZLIT0[7],           ; CHECK TAIL FOR QUAD ALIGNMENT
U 0D12, 1981,8C12,0A30,3847,00CC,8  ;21665          WX.EQ.0?,PATCH                   ;
                                    ;21666
                                    ;21667 =00      ;00################################; BAD ALIGNMENT, RESERVED OPERAND.
                                    ;21668          VA_M[TEMP4],SET MM.NOINT,        ; LOAD ADDRESS OF HEADER
U 0CC8, 0460,4002,403D,84A7,00FF,8  ;21669          NEXT/IE.OPER.FAULT               ; DISABLE MEMORY MANAGEMENT TAKING INTS
                                    ;21670                                           ; ** replaced in PCS **
                                    ;21671
                                    ;21672          ;01################################; ** replaced in PCS **
                                    ;21673          R[TEMP5]_M[TEMP4]-Q,             ; SAVE HEADER - ENTRY
                                    ;21674          PROBE WRITE?,SIZE[IDEP],         ;
;????Validity failure
U 0CC9, 0484,4008,07B1,405D,0D68,4 479*  ;21675     PUSH,NEXT/MM.PRB.WRITE.SIZ       ; CHECK THAT TAIL CAN BE WRITTEN
                                    ;21676
                                    ;21677          ;10################################; ** replaced in PCS **
U 0CCA, 0480,003A,403D,84A7,00D1,3  ;21678          VA_Q,NEXT/CI.PATCH.RET           ; LOAD ADDRESS OF ENTRY
                                    ;21679 =
                                    ;21680 28C8:    ;00********************************; BAD ALIGNMENT, RESERVED OPERAND.
                                    ;21681          VA_M[TEMP4],SET MM.NOINT,        ; LOAD ADDRESS OF HEADER
U 28C8, 0460,4002,403D,84A7,00FF,8  ;21682          NEXT/IE.OPER.FAULT               ; DISABLE MEMORY MANAGEMENT TAKING INTS
                                    ;21683                                           ; ** code in PCS **
                                    ;21684
                                    ;21685 28C9:    ;01********************************; ** code in PCS **
U 28C9, 0484,4008,0031,4047,028C,A  ;21686          R[TEMP5]_M[TEMP4]-Q              ; SAVE HEADER - ENTRY
                                    ;21687
                                    ;21688 28CA:    ;********************************; ** code in PCS **
                                    ;21689          PROBE WRITE?,SIZE[IDEP],         ;
U 28CA, 0480,0036,47B0,005D,0D68,4 479*  ;21690     PUSH,NEXT/MM.PRB.WRITE.SIZ       ; CHECK THAT TAIL CAN BE WRITTEN
                                    ;21691
                                    ;21692 28CB:    ;10********************************; ** code in PCS **
U 28CB, 0480,003A,403D,84A7,00D1,3  ;21693          VA_Q                             ; LOAD ADDRESS OF ENTRY
                                    ;21694
```

```
                                        ;21695 CI.PATCH.RET:
                                        ;21696          ;--------------------------------;
                                        ;21697          WB_M[VA].AND.ZLIT0[7],          ; CHECK ENTRY FOR QUAD ALIGNMENT
U 0D13, 0161,BC12,0A30,3847,00CE,0      ;21698          SET MM.NOINT,WX.EQ.0?           ; DISABLE MEMORY MANAGEMENT TAKING INTS
                                        ;21699
                                        ;21700 =0       ;0-------------------------------;
                                        ;21701          VA_M[TEMP4],                    ; LOAD ADDRESS OF HEADER
U 0CE0, 0C80,4002,403D,84A7,00FF,8      ;21702          NEXT/IE.OPER.FAULT              ;
                                        ;21703
                                        ;21704          ;1-------------------------------;
U 0CE1, 0C80,5592,4020,05D8,00D1,4      ;21705          WRITE M[TEMP5],SIZE[LONG]       ; SET FORWARD LINK IN ENTRY
                                        ;21706
                                        ;21707          ;--------------------------------;
U 0D14, 0C80,0036,4030,0447,00D1,5      ;21708          VA_VA+4                         ;
                                        ;21709
                                        ;21710          ;--------------------------------;
                                        ;21711          WRITE,SIZE[LONG],               ; SET BACKWARD LINK IN ENTRY
U 0D15, 0481,2001,1021,45D8,00D1,6      ;21712          Q_M[MDR]+R[TEMP5]               ;
                                        ;21713
                                        ;21714          ;--------------------------------;
                                        ;21715          WB_M[MDR],                      ; CHECK IF FIRST ENTRY IN QUEUE
U 0D16, 0481,2592,4A70,0047,00CE,2      ;21716          WX.NE.0?                        ;
                                        ;21717
                                        ;21718 =0       ;0-------------------------------; SAVE FORWARD LINK TO ENTRY
                                        ;21719          M[TEMP1]_R[ZERO]-Q,             ; WHICH WILL BE STORED LATER IN HEADER
U 0CE2, 0886,1038,003D,8047,00D1,8      ;21720          NEXT/QU.INSQTI.10               ;
                                        ;21721
                                        ;21722          ;1-------------------------------;
U 0CE3, 0881,2001,0031,04A7,00D1,7      ;21723          VA_M[MDR]+R[TEMP4]              ; LOAD ADDRESS OF TAIL
                                        ;21724
                                        ;21725          ;--------------------------------;
U 0D17, 0880,0038,002D,85D8,00D1,8      ;21726          WRITE -Q,SIZE[LONG]             ; SET FORWARD LINK IN OLD TAIL TO ENTRY
                                        ;21727
                                        ;21728 QU.INSQTI.10:
                                        ;21729          ;--------------------------------;
U 0D18, 0580,4C11,0030,24A7,00D1,9      ;21730          VA_M[TEMP4]+ZLIT0[4]            ; LOAD ADDRESS OF HEADER + 4
                                        ;21731
                                        ;21732          ;--------------------------------;
                                        ;21733          WRITE R[ZERO]-M[TEMP5],         ; SET BACKWARD LINK IN HEADER
U 0D19, 0480,5003,002D,85D8,00D2,C      ;21734          SIZE[LONG],NEXT/QU.UNLOCK       ; GO RELEASE SECONDARY INTERLOCK.
```

```
                                ;21735 .TOC "  Misc. and Queue                    : REMQHI"
                                ;21736
                                ;21737 ;*****************************************************************************
                                ;21738 ;       REMQHI          header.ab, addr.wl
                                ;21739 ;
                                ;21740 ;       Input           VA              addr
                                ;21741 ;                       Q               header
                                ;21742 ;
                                ;21743 ;       Resources       TEMP1           (header)
                                ;21744 ;                       TEMP2           header - entry
                                ;21745 ;                       TEMP4           header (needed to release lock on fault)
                                ;21746 ;                       TEMP5           entry
                                ;21747 ;                       D               successor
                                ;21748 ;
                                ;21749 ;       Subroutines     MM.PRB.WRITE.SIZ
                                ;21750 ;                       QU.REMQHI.20
                                ;21751 ;                       QU.SET.LOCK
                                ;21752 ;
                                ;21753 ;*****************************************************************************
                                ;21754
                                ;21755 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;21756 QU.REMQHI.REG:
                                ;21757         ;-------------------------------; REGISTER MODE ENTRY POINT
U 03AC, 0981,2C1C,7020,0CA7,002B,A ;21758      Q_M[MDR] VA_ZLITO[1]            ; CLOBBER MDR TO AVOID RESERVED OPERAND.
                                ;21759 =0
                                ;21760 QU.REMQHI:
                                ;21761         ;0-------------------------------; MOVE DESTINATION ADDRESS TO Q AND
                                ;21762         VA_R[TEMP4]_Q Q_M[VA],          ; LOAD ADDRESS OF HEADER AND SAVE IT.
                                ;21763         SET FLAG1,                      ; SET FOR USE BY QU.SET.LOCK
U 02BA, 044D,B00C,7031,04A7,04D2,9 ;21764      PUSH,NEXT/QU.SET.LOCK           ; SET THE SECONDARY INTERLOCK.
                                ;21765
                                ;21766         ;1-------------------------------; HEAD = HEADER + (HEADER)
                                ;21767         CLEAR FLAG0,                    ; SET BY QU.SET.LOCK FOR CC SETTING
U 02BB, 0800,4001,2030,44A7,00D1,A ;21768      VA_D M[TEMP4]+R[TEMP1]          ; LOAD ADDRESS OF HEAD AND SAVE IT.
                                ;21769 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                                ;21770         ;-------------------------------;
                                ;21771         WB_M[TEMP1],                    ; CHECK IF QUEUE IS EMPTY
U 0D1A, 0080,1592,4A70,0047,00CE,4 ;21772      WX.NE.0?                        ;
                                ;21773
                                ;21774 =0      ;0-------------------------------;
                                ;21775         R[TEMP2]_D,                     ; SAVE ADDRESS OF HEADER AS NEW HEAD
U 0CE4, 0484,05B2,4030,8047,00D1,C ;21776      NEXT/QU.REMQHI.10               ;
                                ;21777
                                ;21778         ;1-------------------------------;
                                ;21779         READ,SIZE[LONG],                ; GET FORWARD LINK FROM HEAD AND MOVE
U 0CE5, 0884,05B2,4020,8050,00D1,B ;21780      R[TEMP2]_D                      ; ADDRESS OF HEAD FOR NEXT CALCULATION
```

```
                                        ;21781          ;-------------------------------; NEW HEAD = HEAD + (HEAD)
U 0D1B, 0485,2001,0030,8047,00D1,C      ;21782          R[TEMP2]_M[MDR]+RB              ; SAVE ADDRESS OF NEW HEAD
                                        ;21783
                                        ;21784 QU.REMQHI.10:
                                        ;21785          ;-------------------------------;
                                        ;21786          WB_M[TEMP2].AND.ZLIT0[7],       ; CHECK NEW HEAD FOR QUAD ALIGNMENT
U 0D1C, 0D80,2C12,0A30,3847,00CE,6      ;21787          WX.EQ.0?                        ;
                                        ;21788
                                        ;21789 =0       ;0-------------------------------; BAD ALIGNMENT, RESERVED OPERAND
                                        ;21790          VA_M[TEMP4],SET MM.NOINT,       ; LOAD ADDRESS OF HEADER
U 0CE6, 0460,4002,403D,84A7,00FF,8      ;21791          NEXT/IE.OPER.FAULT              ; DISABLE MEMORY MANAGEMENT TAKING INTS
                                        ;21792
                                        ;21793          ;1-------------------------------;
U 0CE7, 0580,2C11,0030,24A7,00CE,8      ;21794          VA_M[TEMP2]+ZLIT0[4]            ; LOAD ADDRESS OF NEW HEAD + 4
                                        ;21795
                                        ;21796 =0       ;0-------------------------------;
                                        ;21797          PROBE WRITE?,SIZE[IDEP],        ; CHECK NEW HEAD + 4 FOR WRITE ACCESS
U 0CE8, 0480,0036,47B0,005D,0D68,4 479* ;21798          PUSH,NEXT/MM.PRB.WRITE.SIZ      ;
                                        ;21799
                                        ;21800          ;1-------------------------------; DISABLE MEMORY MANAGEMENT TAKING INTS
U 0CE9, 0460,003A,403D,84A7,00D1,D      ;21801          VA_Q,SET MM.NOINT               ; LOAD DESTINATION ADDRESS
                                        ;21802
                                        ;21803          ;-------------------------------;
                                        ;21804          R[DST.R]_D,
U 0D1D, 0084,05B2,402C,45DA,00D1,E      ;21805          WRITE NOTREG,SIZE[LONG]         ; WRITE DESTINATION POINTER TO ENTRY
                                        ;21806
                                        ;21807          ;-------------------------------;
U 0D1E, 0580,2C11,0030,24A7,00D1,F      ;21808          VA_M[TEMP2]+ZLIT0[4]            ; LOAD ADDRESS OF NEW HEAD + 4
                                        ;21809
                                        ;21810          ;-------------------------------;
                                        ;21811          CLEAR FLAG2,                    ; CLEAR HERE, THEN SET IF LAST ENTRY
                                        ;21812          WRITE,SIZE[LONG],               ; SET BACKWARD LINK IN NEW HEAD
                                        ;21813          R[TEMP2]_M[TEMP4]-RB,           ; SAVE FOR CALCULATING HEADER LINK
U 0D1F, 0014,4000,0A20,85D8,08CE,A 345* ;21814          WX.EQ.0?                        ; CHECK IF REMOVED LAST ENTRY
                                        ;21815
                                        ;21816 =0
                                        ;21817 QU.REMQHI.20:
                                        ;21818          ;0-------------------------------; DID NOT REMOVE LAST ENTRY FROM QUEUE
                                        ;21819          M[TEMP1]_-R[TEMP2],             ; SAVE HEADER FORWARD LINK
U 0CEA, 0486,15BF,0030,8047,00D2,C      ;21820          NEXT/QU.UNLOCK                  ;
                                        ;21821
                                        ;21822          ;1-------------------------------; REMOVED LAST ENTRY FROM QUEUE
                                        ;21823          M[TEMP1]_-R[TEMP2],             ; SAVE HEADER FORWARD LINK
U 0CEB, 0C56,15BF,0030,8047,00D2,C      ;21824          SET FLAG2,NEXT/QU.UNLOCK        ; PSL<Z> <- 1
```

```
                                ;21825 .TOC "  Misc. and Queue                  : REMQTI"
                                ;21826
                                ;21827 ;************************ ************************************************************
                                ;21828 ;       REMQTI          header.ab, addr.wl
                                ;21829 ;
                                ;21830 ;       Input           VA              addr
                                ;21831 ;                       Q               header
                                ;21832 ;
                                ;21833 ;       Resources       TEMP1           (header)
                                ;21834 ;                       TEMP2           header - entry
                                ;21835 ;                       TEMP4           header (needed to release lock on fault)
                                ;21836 ;                       TEMP5           entry
                                ;21837 ;                       D               successor
                                ;21838 ;
                                ;21839 ;       Subroutines     MM.PRB.WRITE.SIZ
                                ;21840 ;                       QU.REMQHI.20
                                ;21841 ;                       QU.SET.LOCK
                                ;21842 ;
                                ;21843 ;*************************************************************************************
                                ;21844
                                ;21845 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;21846 QU.REMQTI.REG:
                                ;21847         ;-------------------------------; REGISTER MODE ENTRY POINT
U 03AD, 0181,2C1C,7020,0CA7,002C.A ;21848      Q_M[MDR] VA_ZLIT0[1]            ; CLOBBER MDR TO AVOID RESERVED OPERAND.
                                ;21849 =0
                                ;21850 QU.REMQTI:
                                ;21851         ;0-------------------------------; MOVE DESTINATION ADDRESS TO Q AND
                                ;21852         VA_R[TEMP4]_Q Q_M[VA],          ; LOAD ADDRESS OF HEADER AND SAVE IT.
                                ;21853         SET FLAG1,                      ; SET FOR USE BY QU.SET.LOCK
U 02CA, 044D,B00C,7031,04A7,04D2,9 ;21854      PUSH.NEXT/QU.SET.LOCK           ; SET THE SECONDARY INTERLOCK.
                                ;21855
                                ;21856         ;1-------------------------------;
                                ;21857         CLEAR FLAG0,                    ; SET BY QU.SET.LOCK FOR CC SETTING
U 02CB, 0D00,4C11,0030,24A7,00D2,0 ;21858      VA_M[TEMP4]+ZLIT0[4]            ; LOAD ADDRESS OF HEADER + 4
                                ;21859 .REGION/MISQUE.R1L,MISQUE.R1H/MISQUE.R2L,MISQUE.R2H/MISQUE.R3L,MISQUE.R3H
                                ;21860         ;-------------------------------;
U 0D20, 0880,0036,4020,0050,00D2,1 ;21861      READ,SIZE[LONG]                 ; FETCH LINK TO TAIL
                                ;21862
                                ;21863         ;-------------------------------;
U 0D21, 0C81,2001,2031,0047,00D2,2 ;21864      D_M[MDR]+R[TEMP4]               ; SAVE ADDRESS OF TAIL
                                ;21865
                                ;21866         ;-------------------------------;
                                ;21867         WB_D.AND.ZLIT0[7],              ; CHECK TAIL FOR QUAD ALIGNMENT
U 0D22, 0D80,0C32,0A30,3847,00CE,C ;21868      WX.EQ.0?                        ;
```

```
                                        ;21869 =0     ;0------------------------------;
                                        ;21870        VA_M[TEMP4],SET MM.NOINT,       ; LOAD ADDRESS OF HEADER
U OCEC, 0460,4002,403D,84A7,00FF,8      ;21871        NEXT/IE.OPER.FAULT              ; DISABLE MEMORY MANAGEMENT TAKING INTS
                                        ;21872
                                        ;21873        ;1------------------------------;
U OCED, 0580,0C31,0030,24A7,00D2,3      ;21874        VA_D+ZLITO[4]                   ; LOAD ADDRESS OF LINK TO NEW TAIL
                                        ;21875
                                        ;21876        ;------------------------------;
                                        ;21877        READ,SIZE[LONG],                ; FETCH LINK TO NEW TAIL
U OD23, 0884,05B2,4020,8050,00D2,4      ;21878        R[TEMP2]_D                      ; MOVE D TO TEMP(FOR LATER CALCULATION)
                                        ;21879
                                        ;21880        ;------------------------------;
U OD24, 0485,2001,0030,84A7,00CE,E      ;21881        VA_R[TEMP2]_M[MDR]+RB           ; COMPUTE ADDRESS OF NEW TAIL
                                        ;21882
                                        ;21883 =0     ;0------------------------------;
                                        ;21884        PROBE WRITE?,SIZE[IDEP],        ; CHECK THAT NEW TAIL CAN BE WRITTEN
U OCEE, 0480,0036,47B0,005D,0D68,4 479* ;21885        PUSH,NEXT/MM.PRB.WRITE.SIZ      ;
                                        ;21886
                                        ;21887        ;1------------------------------;
                                        ;21888        WB_M[TEMP2].AND.ZLITO[7],        ; CHECK THAT NEW TAIL IS QUAD ALIGNED
U OCEF, 0580,2C12,0A30,3847,00CF,0      ;21889        WX.EQ.0?                        ;
                                        ;21890
                                        ;21891 =0     ;0------------------------------; BAD ALIGNMENT, RESERVED OPERAND
                                        ;21892        VA_M[TEMP4],SET MM.NOINT,       ; LOAD ADDRESS OF HEADER
U OCF0, 0460,4002,403D,84A7,00FF,8      ;21893        NEXT/IE.OPER.FAULT              ; DISABLE MEMORY MANAGEMENT TAKING INTS
                                        ;21894
                                        ;21895        ;1------------------------------; DISABLE MEMORY MANAGEMENT TAKING INTS
U OCF1, 0C60,003A,403D,84A7,00D2,5      ;21896        VA_Q,SET MM.NOINT               ; LOAD ADDRESS OF DESTINATION
                                        ;21897
                                        ;21898        ;------------------------------;
                                        ;21899        R[DST.R]_D,                     ;
U OD25, 0884,05B2,402C,45DA,00D2,6      ;21900        WRITE NOTREG,SIZE[LONG]         ; WRITE DESTINATION POINTER TO ENTRY
                                        ;21901
                                        ;21902        ;------------------------------;
U OD26, 0D80,4C11,0030,24A7,00D2,7      ;21903        VA_M[TEMP4]+ZLITO[4]            ; LOAD ADDRESS OF HEADER + 4
                                        ;21904
                                        ;21905        ;------------------------------;
                                        ;21906        WRITE R[TEMP2]-M[TEMP4],        ; SET BACKWARD LINK IN HEADER
U OD27, 0080,4003,0A60,85D8,08CF,2 339* ;21907        SIZE[LONG],WX.NE.0?             ; CHECK IF REMOVED LAST ENTRY
                                        ;21908
                                        ;21909 =0     ;0------------------------------;
                                        ;21910        M[TEMP1]_ZLITO[0],              ; QUEUE IS EMPTY, ZERO HEADER
                                        ;21911        SET FLAG2,                      ;
U OCF2, 0156,1C37,0030,0047,00D2,C      ;21912        NEXT/QU.UNLOCK                  ;
                                        ;21913
                                        ;21914        ;1------------------------------;
                                        ;21915        CLEAR FLAG2,                    ;
U OCF3, 0C10,2002,403D,84A7,00D2,8      ;21916        VA_M[TEMP2]                     ; LOAD ADDRESS OF NEW TAIL
                                        ;21917
                                        ;21918        ;------------------------------;
                                        ;21919        WRITE R[TEMP4]-M[TEMP2],        ; SET FORWARD LINK IN NEW TAIL
U OD28, 0480,2003,0021,05D8,00D2,C      ;21920        SIZE[LONG],NEXT/QU.UNLOCK       ;
```

```
                        ;21921 .TOC "  Misc. and Queue                  : QU.SET.LOCK"
                        ;21922
                        ;21923 ;*****************************************************************************
                        ;21924 ;       Entry points    QU.LOCK
                        ;21925 ;
                        ;21926 ;       Input           VA              Address of header
                        ;21927 ;                       TEMP4           Address of header
                        ;21928 ;                       Q               If INSQXI then address of entry
                        ;21929 ;                                       If REMQXI then address of destination
                        ;21930 ;                       FLAG1           If INSQXI then clear
                        ;21931 ;                                       If REMQXI then set
                        ;21932 ;
                        ;21933 ;       Resources       MDR             Read header
                        ;21934 ;
                        ;21935 ;       Output          TEMP1           (header)
                        ;21936 ;                       FPDOFFSET       19.
                        ;21937 ;                       FPD             Set
                        ;21938 ;                       FLAG1           Set if queue is empty else clear
                        ;21939 ;                       FLAG2           Set if queue is empty else clear
                        ;21940 ;                       SOFTWARE INTERLOCK      Set
                        ;21941 ;
                        ;21942 ;       QU.LOCK checks that "header"  is quad aligned and not equal to
                        ;21943 ;       "addr" or "entry" respectively for REMQXI and INSQXI.
                        ;21944 ;       Then the header is read and the software interlock checked and
                        ;21945 ;       set if clear.
                        ;21946 ;
                        ;21947 ;*****************************************************************************
                        ;21948
                        ;21949 QU.SET.LOCK:
                        ;21950                  ;-------------------------------;
                        ;21951          WB_M[TEMP4 .AND.ZLIT0[7],       ; CHECK HEADER FOR QUAD ALIGNMENT
U 0D29, 0D80,4C12,0A30, ;21952          WX.EQ.0?                        ;
                        ;21953
                        ;21954 =0       ;0-------------------------------;
U 0CF4, 0C80,0036,4030, ;21955          NEXT/IE.OPER.FAULT              ; TAKE A RESRVED OPERAND FAULT
                        ;21956
                        ;21957          ;1-------------------------------;
                        ;21958          WB_M[TEMP4].XOR.Q,              ; CHECK HEADER .NE. (ADDR OR ENTRY)
U 0CF5, 0080,400B,4A70, ;21959          WX.NE.0?                        ; (ADDR IF INSQ, ENTRY IF REMQ)
                        ;21960
                        ;21961 =0       ;0-------------------------------;
U 0CF6, 0C80,0036,4030, ;21962          NEXT/IE.OPER.FAULT              ; TAKE A RESRVED OPERAND FAULT
                        ;21963
                        ;21964          ;1-------------------------------;
U 0CF7, 0080,0036,4020, ;21965          READ.MOD.LOCK,SIZE[LONG]        ; ATTEMPT TO SET SECONDARY LOCK
                        ;21966
                        ;21967          ;*********************************; ** jump to PCS **
                        ;21968          M[FPDOFFSET]_ZLIT0[19.],SET FPD,; SETUP TO RELEASE LOCK IF FAULT OCCURS
U 0D2A, 19EE,CC37,0030, ;21969          PATCH
```

```
                                        ;21970          ;################################;** replaced in PCS **
                                        ;21971          SET FLAG0,                      ;SET INCASE LOCKED OUT
                                        ;21972          R[TEMP1] M[MDR],WB<1-0>?,       ;SEE IF LOCK ALREADY SET
U 0D2B, 0845,2592,4270,4047,0131,2      ;21973          NEXT/QU.LOCK.CHECK              ;
                                        ;21974
                                        ;21975 =10
                                        ;21976 QU.LOCKED.ALIGN:                         ;** replaced in PCS **
                                        ;21977          ;10##############################;SOFTWARE LOCK SET,WB<2-1> NOT ZERO
                                        ;21978          WB_M[TEMP1].AND.ZLIT0[7],       ;THIS SHOULD GO TO IE.OPER.FAULT
U 0CCE, 0D80,1C12,0A30,3847,00CF,8      ;21979          WX.EQ.0?,NEXT/QU.CHECK.ALIGN    ;BUT THIS WAY SAVE MODIFYING THIS ROW
                                        ;21980
                                        ;21981
                                        ;21982          ;11##############################;SOFTWARE LOCK ALREADY SET
                                        ;21983          R[TEMP2] FLAGS,CLEAR FPD,       ;PREPARE TO SET CONDITION CODES
U 0CCF, 0CE4,0036,4030,8387,00D2,E      ;21984          NEXT/QU.UNLOCK.10               ;** replaced in PCS **
                                        ;21985 =0
                                        ;21986 QU.CHECK.ALIGN:
                                        ;21987          ;0##############################;** replaced in PCS **
                                        ;21988          VA_M[TEMP4],SET MM.NOINT,       ;LOAD ADDRESS OF HEADER
U 0CF8, 0460,4002,403D,84A7,00FF,8      ;21989          NEXT/IE.OPER.FAULT              ;DISABLE MEMORY MANAGEMENT TAKING INTS
                                        ;21990
                                        ;21991 QU.CHECK.ALIGN.1:
                                        ;21992          ;1------------------------------;
                                        ;21993          SET FLAG1,                      ;SEE HEADER COMMENT
U 0CF9, 0448,1592,4A30,0047,08CF,A 322* ;21994          WB_M[TEMP1],WX.EQ.0?            ;CHECK IF QUEUE IS EMPTY TO SET CC'S
                                        ;21995 =0
                                        ;21996          ;0------------------------------;RELEASE HARDWARE LOCK, AND SET
                                        ;21997          WRITE.UL M[TEMP1].OR.ZLIT0[1],  ;SOFTWARE LOCK
                                        ;21998          CLEAR FLAG1,                    ;SEE HEADER COMMENT
U 0CFA, 0508,1C12,40A0,0DDB,0000,1      ;21999          SIZE[LONG],RETURN [+1]          ;
                                        ;22000
                                        ;22001          ;1------------------------------;RELEASE HARDWARE LOCK, AND SET
                                        ;22002          WRITE.UL M[TEMP1].OR.ZLIT0[1],  ;SOFTWARE LOCK, SET PSL<Z> FOR REMQXI
                                        ;22003          SIZE[LONG],                     ;
U 0CFB, 0D50,1C12,40A0,0DD5,0000,1      ;22004          SET FLAG2,RETURN [+1]           ;NO ENTRY TO REMOVE
                                        ;22005
```

```
                                        ;22006 292B:  ;================================; ** u-code in PCS **
                                        ;22007        SET FLAGO,                        ; SET INCASE LOCKED OUT
  292B, 0045,2592,4030,4047,0292,C      ;22008        R[TEMP1]_M[MDR]                   ; SEE IF LOCK ALREADY SET
                                        ;22009
                                        ;22010 292C:  ;================================; ** u-code in PCS **
                                        ;22011        WB_M[TEMP1].AND.ZLIT0[6],         ; CHECK IF (HEADER)<2-1> BITS ARE ZERO
U 292C, 0980,1C12,0A30,3047,0A93,2 335* ;22012        WX.EQ.0?
                                        ;22013
                                        ;22014 2932:  ;0===============================; RESERVED OPERAND, TAKE FAULT EXIT
                                        ;22015        WRITE.UL M[TEMP1],SIZE[LONG],     ; RELEASE HAREWARE LOCK
                                        ;22016        CLEAR FPD,                        ; ** u-code in PCS **
U 2932, 08E0,1592,4020,05DB,00FF,8      ;22017        NEXT/IE.OPER.FAULT                ;
                                        ;22018
                                        ;22019 2933:  ;1===============================; ** u-code in PCS **
                                        ;22020        WB_M[TEMP1].AND.ZLIT0[1],         ; CHECK IF 2ND INTERLOCK BIT SET
U 2933, 0180,1C12,0A30,0847,0A93,4 335* ;22021        WX.EQ.0?                          ;
                                        ;22022
                                        ;22023 2934:  ;0===============================; 2ND INTERLOCK IS SET
                                        ;22024        R[TEMP2]_FLAGS,CLEAR FPD,         ; PREPARE TO SET CONDITION CODES
U 2934, 0CE4,0036,4030,8387,00D2,E      ;22025        NEXT/QU.UNLOCK.10                 ; ** u-code in PCS **
                                        ;22026
                                        ;22027 2935:  ;1===============================; 2ND INTERLOCK ISN'T SET
U 2935, 0480,0036,4030,0047,00CF,9      ;22028        NEXT/QU.CHECK.ALIGN.1             ; ** u-code in PCS **
                                        ;22029
```

```
                                ;22030 .TOC "  Misc. and Queue                  : QU.UNLOCK"
                                ;22031
                                ;22032 ;*****************************************************************************
                                ;22033 ;       Entry point     QU.UNLOCK
                                ;22034 ;
                                ;22035 ;       Input           TEMP1           New contents of header
                                ;22036 ;                       TEMP4           Address of header
                                ;22037 ;
                                ;22038 ;       QU.UNLOCK is the normal exit for REMQXI and INSQXI.
                                ;22039 ;       QU.UNLOCK.10 is branched to if lsb of header already set
                                ;22040 ;
                                ;22041 ;*****************************************************************************
                                ;22042
                                ;22043
                                ;22044 QU.UNLOCK:
                                ;22045          ;-------------------------------;
U 0D2C, 0CE0,4002,403D,84A7,00D2,D ;22046       VA_M[TEMP4],CLEAR FPD           ; LOAD ADDRESS OF HEADER
                                ;22047
                                ;22048          ;-------------------------------;
                                ;22049          READ.MOD.LOCK,SIZE[LONG],       ;
U 0D2D, 0884,0036,4020,8393,00D2,E ;22050       R[TEMP2]_FLAGS                  ; PREPARE TO SET CONDITION CODES.
                                ;22051
                                ;22052 QU.UNLOCK.10:                            ; THIS IS AN ENTRY FOR SOFTWARE
                                ;22053          ;-------------------------------; INTERLOCK ALREADY SET
U 0D2E, 0C80,2592,4030,00A7,00D2,F ;22054       CC_M[TEMP2]
                                ;22055
                                ;22056          ;-------------------------------;
                                ;22057          WRITE.UL M[TEMP1],SIZE[LONG],   ; RELEASE BOTH INTERLOCKS
U 0D2F, 0480,1592,4120,05DB,003F,9 ;22058       IRD1                            ;
                                ;22059
                                ;22060
                                ;22061          ;++++++++++++++++++++++++++++++
                                ;22062          ; the following 2 locations are in decmal.mic module
                                ;22063          ;++++++++++++++++++++++++++++++
                                ;22064 ;
                                ;22065 ;1296:   ;++++++++ force address ++++++++;
                                ;22066 ;QU.RESV.OPER:                           ; CAN'T COMPLETE INTERLOCKED QUE INSTR
                                ;22067 ;
                                ;22068 ;        ;-------------------------------; SAVE FAULTING ADDRESS INCASE
                                ;22069 ;        M[TEMP6]_R[MM.TEMP3]            ; IN CASE OF A TB-MISS ON HEADER
                                ;22070 ;
                                ;22071 ;1297:   ;+++++++ force address ++++++++*;
                                ;22072 ;        ;-------------------------------; GET SET TO READ HEADER
                                ;22073 ;        READ.MOD.LOCK,SIZE[LONG],       ; WITH HARDWARE INTERLOCK IN EFFECT
                                ;22074 ;        CLEAR FPD,                      ;
                                ;22075 ;        NEXT/QU.RESV.FIX                ;REMAINDER IN DECMAL MODULE
                                ;22076
                                ;22077
                                ;22078 ;1052:   ;+++++++ force address +++++++++*;
                                ;22079 ;QU.RESV.FIX:
                                ;22080 ;        ;-------------------------------;
                                ;22081 ;        WRITE.UL M[TEMP1].ANDNOT.ZLIT0[1],      ; CLEAR SOFT LOCK BIT
                                ;22082 ;        SIZE[LONG]                              ; AND UNLOCK HARDWARE INTERLOCK
                                ;22083 ;
                                ;22084 ;1066:   ;+++++++ force address +++++++++*;
```

```
                                ;22085 ;       ;--------------------------------;
                                ;22086 ;       R[MM.TEMP3] M[TEMP6],            ; RECOVER ORIGINAL FAULTING ADR
                                ;22087 ;       NEXT/QU.RESV.EXIT                ; RETURN TO EXCEPTION HANDLER FLOW
                                ;22088
                                ;22089 FREE.0D30:
                                ;22090         ;+++++++ force address ++++++++*;
                                ;22091         ;--------------------------------;
                                ;22092         WRITE.UL M[TEMP1],SIZE[LONG],    ; RELEASE BOTH INTERLOCKS
U 0D30, 00E0,1592,4020,05DB,00BF,D ;22093      CLEAR FPD                        ;
                                ;22094
                                ;22095 0BFD:
                                ;22096 QU.RESV.EXIT:
                                ;22097         ;+++++++ force address ++++++++*;
                                ;22098         R[TEMP0] RBSP,RBSP.EQ.0?,PUSH,   ; ROLL BACK SIDE EFFECTS
U 0BFD, 0885,E036,4BB0,0047,04F5,A ;22099      NEXT/IE.RBS.RBACK                ;
                                ;22100
                                ;22101 0BFF:
                                ;22102         ;+++++++ force address ++++++++*;
U 0BFF, 0480,0036,4030,0047,00FE,2 ;22103      NEXT/IE.PACK.DONE                ; INCASE A TB-MISS HAD OCCURRED
```

```
                                        ;22104 .TOC "  Misc. and Queue                  : Ird Rom Definition"
;22105  .NOBIN
;22106
;22107  .ICODE  ;       OPS             REG                     MEM                OPS         FPA REG                 FPA MEM
;22108       0B9: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                    ;BICPSW
;22109            IRD1[LOD][OS.RED                ]                          [LOD][OS.RED                ]
;22110  .OCODE
;22111       0B9: CNT0[NOP][MS.BICPSW             ][MS.BICPSW             ]  [NOP][MS.BICPSW             ][MS.BICPSW             ],
;22112            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;22113
;22114  .ICODE
;22115       0B8: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                    ;BISPSW
;22116            IRD1[LOD][OS.RED                ]                          [LOD][OS.RED                ]
;22117  .OCODE
;22118       0B8: CNT0[NOP][MS.BISPSW             ][MS.BISPSW             ]  [NOP][MS.BISPSW             ][MS.BISPSW             ],
;22119            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;22120
;22121  .ICODE
;22122       003: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                    ;BPT
;22123            IRD1[NOP][MS.BPT                ]                          [NOP][MS.BPT                ]
;22124  .OCODE
;22125       003: CNT0[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ],
;22126            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;22127
;22128  .ICODE
;22129       000: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                    ;HALT
;22130            IRD1[NOP][MS.HALT               ]                          [NOP][MS.HALT               ]
;22131  .OCODE
;22132       000: CNT0[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ],
;22133            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;22134
;22135  .ICODE
;22136       00A: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                    ;INDEX
;22137            IRD1[LOD][OS.RED                ]                          [LOD][OS.RED                ]
;22138  .OCODE
;22139       00A: CNT0[LOD][OS.RED                ][OS.RED                ]  [LOD][OS.RED                ][OS.RED                ],
;22140            CNT1[NOP][MS.INDEX              ][MS.INDEX              ]  [NOP][MS.INDEX              ][MS.INDEX              ]
;22141
;22142  .ICODE
;22143       05C: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                    ;INSQHI
;22144            IRD1[LOD][OS.ADD                ]                          [LOD][OS.ADD                ]
;22145  .OCODE
;22146       05C: CNT0[LOD][OS.ADD                ][OS.ADD                ]  [LOD][OS.ADD                ][OS.ADD                ],
;22147            CNT1[NOP][QU.INSQHI             ][QU.INSQHI             ]  [NOP][QU.INSQHI             ][QU.INSQHI             ]
;22148
;22149  .ICODE
;22150       05D: FPD [NOP][IE.OPCOD.DEC          ]                          [NOP][IE.OPCOD.DEC          ],                    ;INSQTI
;22151            IRD1[LOD][OS.ADD                ]                          [LOD][OS.ADD                ]
;22152  .OCODE
;22153       05D: CNT0[LOD][OS.ADD                ][OS.ADD                ]  [LOD][OS.ADD                ][OS.ADD                ],
;22154            CNT1[NOP][QU.INSQTI             ][QU.INSQTI             ]  [NOP][QU.INSQTI             ][QU.INSQTI             ]
```

```
;22155  .ICODE  ;       OPS             REG                     MEM                OPS         FPA REG                 FPA MEM
;22156       00E: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                  ;INSQUE
;22157            IRD1[LOD][OS.ADD                ]                                [LOD][OS.ADD                ]
;22158  .OCODE
;22159       00E: CNT0[LOD][OS.ADD                ][OS.ADD                ]  [LOD][OS.ADD                ][OS.ADD                ],
;22160            CNT1[NOP][QU.INSQUE             ][QU.INSQUE             ]  [NOP][QU.INSQUE             ][QU.INSQUE             ]
;22161
;22162  .ICODE
;22163       0DC: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                  ;MOVPSL
;22164            IRD1[LOD][OS.WRT1               ]                                [LOD][OS.WRT1               ]
;22165  .OCODE
;22166       0DC: CNT0[NOP][MS.MOVPSL             ][MS.MOVPSL             ]  [NOP][MS.MOVPSL             ][MS.MOVPSL             ],
;22167            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;22168
;22169  .ICODE
;22170       001: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                  ;NOP
;22171            IRD1[NOP][MS.NOP                ]                                [NOP][MS.NOP                ]
;22172  .OCODE
;22173       001: CNT0[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ],
;22174            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;22175
;22176  .ICODE
;22177       0BA: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                  ;POPR
;22178            IRD1[LOD][OS.RED                ]                                [LOD][OS.RED                ]
;22179  .OCODE
;22180       0BA: CNT0[NOP][MS.POPR               ][MS.POPR               ]  [NOP][MS.POPR               ][MS.POPR               ],
;22181            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;22182
;22183  .ICODE
;22184       0BB: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                  ;PUSHR
;22185            IRD1[LOD][OS.RED                ]                                [LOD][OS.RED                ]
;22186  .OCODE
;22187       0BB: CNT0[NOP][MS.PUSHR              ][MS.PUSHR              ]  [NOP][MS.PUSHR              ][MS.PUSHR              ],
;22188            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;22189
;22190  .ICODE
;22191       05E: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                  ;REMQHI
;22192            IRD1[LOD][OS.ADD                ]                                [LOD][OS.ADD                ]
;22193  .OCODE
;22194       05E: CNT0[LOD][QU.REMQHI.REG         ][OS.WRT2               ]  [LOD][QU.REMQHI.REG         ][OS.WRT2               ],
;22195            CNT1[NOP][QU.REMQHI             ][QU.REMQHI             ]  [NOP][QU.REMQHI             ][QU.REMQHI             ]
;22196
;22197  .ICODE
;22198       05F: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                  ;REMQTI
;22199            IRD1[LOD][OS.ADD                ]                                [LOD][OS.ADD                ]
;22200  .OCODE
;22201       05F: CNT0[LOD][QU.REMQTI.REG         ][OS.WRT2               ]  [LOD][QU.REMQTI.REG         ][OS.WRT2               ],
;22202            CNT1[NOP][QU.REMQTI             ][QU.REMQTI             ]  [NOP][QU.REMQTI             ][QU.REMQTI             ]
;22203
;22204  .ICODE
;22205       00F: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                  ;REMQUE
;22206            IRD1[LOD][OS.ADD                ]                                [LOD][OS.ADD                ]
;22207  .OCODE
;22208       00F: CNT0[LOD][QU.REMQUE.REG         ][OS.WRT2               ]  [LOD][QU.REMQUE.REG         ][OS.WRT2               ],
;22209            CNT1[NOP][QU.REMQUE.MEM         ][QU.REMQUE.MEM         ]  [NOP][QU.REMQUE.MEM         ][QU.REMQUE.MEM         ]
```

```
;22210  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;22211       OFC: FPD [NOP][MS.XFC                 ]                       [NOP][MS.XFC                 ],          ;XFC
;22212            IRD1[NOP][MS.XFC                 ]                       [NOP][MS.XFC                 ]
;22213  .OCODE
;22214       OFC: CNT0[NOP][FX.IE.FD.RET ][FX.IE.FD.RET ]  [NOP][FX.IE.FD.RET ][FX.IE.FD.RET ],
;22215            CNT1[NOP][FX.IE.FD.RET ][FX.IE.FD.RET ]  [NOP][FX.IE.FD.RET ][FX.IE.FD.RET ]
```

```
;22216  .TOC "  Misc. and Queue                    : Dsize Rom Definition"
;22217  .DCODE
;22218
;22219  0B9:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;BICPSW
;22220
;22221  0B8:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;BISPSW
;22222
;22223  003:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;BPT
;22224
;22225  000:    SIZE [BYTE] [WORD] [LONG] [QUAD] [WORD] [LONG]  ;HALT (used by micro verify)
;22226
;22227  00A:    SIZE [LONG] [LONG] [LONG] [LONG] [LONG] [LONG]  ;INDEX
;22228
;22229  05C:    SIZE [BYTE] [QUAD] [   0] [   0] [   0] [   0]  ;INSQHI
;22230
;22231  05D:    SIZE [BYTE] [QUAD] [   0] [   0] [   0] [   0]  ;INSQTI
;22232
;22233  00E:    SIZE [BYTE] [BYTE] [   0] [   0] [   0] [   0]  ;INSQUE
;22234
;22235  0DC:    SIZE [LONG] [   0] [   0] [   0] [   0] [   0]  ;MOVPSL
;22236
;22237  001:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;NOP
;22238
;22239  0BA:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;POPR
;22240
;22241  0BB:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;PUSHR
;22242
;22243  05E:    SIZE [QUAD] [LONG] [   0] [   0] [   0] [   0]  ;REMQHI
;22244
;22245  05F:    SIZE [QUAD] [LONG] [   0] [   0] [   0] [   0]  ;REMQTI
;22246
;22247  00F:    SIZE [BYTE] [LONG] [   0] [   0] [   0] [   0]  ;REMQUE
;22248
;22249  0FC:    SIZE [LONG] [LONG] [LONG] [LONG] [LONG] [LONG]  ;XFC
;22250
;22251  .UCODE
                                                  ;22252 .BIN
```

22252; This page intentionally left blank.

```
                                        ;22253 .TOC    "CHAR.MIC"
                                        ;22254 .TOC    "REVISION 23.0"
                                        ;22255 ;       Gerard Koeckhoven, John DeRosa, B. WARD
                                        ;22256
;22257  .NOBIN
;22258  .TOC    "       Revision History"
;22259
;22260  ; Rev.  Explanation
;22261  ; 23    Swap a couple un-constrained words with INIT.MIC to make a return -19
;22262  ;       work correctly.
;22263  ; 22    Fix SKPC to read longword aligned
;22264  ;       Fix MATCHC to compare src length with obj length as longwords
;22265  ; 21    Fix MATCHC to not clobber R0
;22266  ; 20    Optimize MATCHC fix
;22267  ; 19    Fix MATCHC to not immediately exit the instruction after it finds the
;22268  ;       SRC length  to be equal to zero.
;22269  ; 18    Fix MATCHC to exit the instruction immediately when the obj length >
;22270  ;       SRC length.
;22271  ; 17    Fix MATCHC FPD bug if bus error/fault in CS.C3 loop.
;22272  ; 16    Initial release.
                                        ;22273         .BIN
```

```
;22274 .TOC     "       Character String                : Operand fetch and Initialization"
;22275
;22276          ;*****************************************************************************
;22277          ;       MOVC3
;22278          ;
;22279          ;       OUTPUT          R0              LENGTH
;22280          ;                       D                  ''
;22281          ;                       Q                  ''
;22282          ;                       TEMP1
;22283          ;                       R1              SOURCE ADDRESS
;22284          ;                       PC                 ''      ''
;22285          ;                       TEMP2                 ''
;22286          ;                       R3              DESTINATION ADDRESS
;22287          ;                       VA                    ''
;22288          ;                       R2              DELTA PC'0(8BITS'24BITS)
;22289          ;                       TEMP6           1(ROTATED LEFT BY 8)
;22290          ;                       CC SET          (R0-R0)
;22291          ;                       FPD             SET
;22292          ;
;22293          ;       RESOURCES       TEMP6           FOR SAVING DELTA PC
;22294          ;
;22295          ;       SUBROUTINES     CS.EXP.FILLER   EXPANDS TEMP5
;22296          ;                       CS.R2.GETS.DELPC.R2   R2_DELTA PC'R2
;22297          ;*****************************************************************************
;22298
;22299          ;*****************************************************************************
;22300          ;       MOVC5
;22301          ;
;22302          ;       OUTPUT          R0              MIN(SRC LGTH,DEST LGTH)
;22303          ;                       Q                  ''    ''         ''
;22304          ;                       D                  ''    ''         ''
;22305          ;                       R1              SOURCE ADDRESS
;22306          ;                       PC                 ''      ''
;22307          ;                       TEMP2                 ''      ''
;22308          ;                       TEMP5           FILL(EXPANDED)
;22309          ;                       TEMP3           DESTINATION LENGTH
;22310          ;                       FLAG1           SET(FOR CMPC5 AND MOVC5)
;22311          ;                       MTEMP10         SRC LGTH - DST LGTH
;22312          ;                       R3              DESTINATION ADDRESS
;22313          ;                       VA                    ''      ''
;22314          ;                       R2              DELTA PC'0(8BITS'24BITS)
;22315          ;                       TEMP6           1(ROTATED LEFT BY 8)
;22316          ;                       CC SET          (TEMP10_R0-TMP3)
;22317          ;                       FPD             SET
;22318          ;
;22319          ;       RESOURCES       TEMP6           FOR SAVING DELTA PC
;22320          ;
;22321          ;       SUBROUTINES     CS.EXP.FILLER   EXPANDS TEMP5
;22322          ;                       CS.R2.GETS.DELPC.R2   R2_DELTA PC'R2
;22323          ;*****************************************************************************
```

```
;22324  ;*************************************************************************
;22325  ;       CMPC3   SAME AS MOVC3 EXCEPT:
;22326  ;
;22327  ;       OUTPUT          CC IS NOT SET
;22328  ;                       Q,R0 DO NOT HAVE LENGTH
;22329  ;                               (R0 WILL GET IT IN CMPC3)
;22330  ;*************************************************************************
;22331
;22332  ;*************************************************************************
;22333  ;       CMPC5   SAME AS MOVC5 EXCEPT:
;22334  ;
;22335  ;       OUTPUT          CC IS NOT SET
;22336  ;*************************************************************************
;22337
;22338  ;*************************************************************************
;22339  ;       MOVTC and MOVTUC
;22340  ;
;22341  ;       OUTPUT          R0              MIN(SRC LGTH,DEST LGTH)
;22342  ;                       Q                  ''      ''          ''
;22343  ;                       D                  ''      ''          ''
;22344  ;                       R1              SOURCE ADDRESS
;22345  ;                       PC                 ''          ''
;22346  ;                       TEMP2              ''          ''
;22347  ;                       TEMP5           ESC(EXPANDED)
;22348  ;                       TEMP7           TABLE ADDRESS
;22349  ;                       TEMP3           DESTINATION LENGTH
;22350  ;                       FLAG0           SET(FOR MOVTC & MOVTUC)
;22351  ;                       FLAG1           SET(FOR CMPC5 & MOVC5)
;22352  ;                       MTEMP10         SRC LGTH - DST LGTH
;22353  ;                       R3              DESTINATION ADDRESS
;22354  ;                       VA                         ''          ''
;22355  ;                       R2              DELTA PC'0(8BITS'24BITS)
;22356  ;                       TEMP6           4(ROTATED LEFT BY 8)
;22357  ;                       FPD             SET
;22358  ;
;22359  ;       RESOURCES       TEMP6           FOR SAVING DELTA PC
;22360  ;
;22361  ;       SUBROUTINES     CS.EXP.FILLER   EXPANDS TEMP5
;22362  ;                       CS.R2.GETS.DELPC.R2   R2_DELTA PC'R2
;22363  ;*************************************************************************
```

```
;22364  ;*************************************************************************
;22365  ;       SPANC and SCANC
;22366  ;       OUTPUT          R0              LENGTH
;22367  ;                       D                 ''
;22368  ;                       Q                 ''
;22369  ;                       R1              ADDRESS
;22370  ;                       PC                ''
;22371  ;                       R3              TABLE ADDRESS
;22372  ;                       TEMP5           MASK
;22373  ;                       R2              ORIGINAL PC
;22374  ;                       FPD             SET
;22375  ;*************************************************************************
;22376  
;22377  ;*************************************************************************
;22378  ;       LOCC and SKPC
;22379  ;
;22380  ;       OUTPUT          TEMP5           CHARACTER
;22381  ;                       R0              LENGTH
;22382  ;                       D                 ''
;22383  ;                       Q               ORIGINAL PC
;22384  ;                       PC              ADDRESS
;22385  ;                       R1                ''
;22386  ;
;22387  ;       SUBROUTINES     CS.EXP.FILLER   EXPANDS TEMP5
;22388  ;*************************************************************************
;22389  
;22390  ;*************************************************************************
;22391  ;       MATCHC
;22392  ;
;22393  ;       OUTPUT          R2              DELTA PC'0'OBJECT LENGTH
;22394  ;                                       (8 BITS)'(8)'(16 BITS)
;22395  ;                       Q                 ''        ''
;22396  ;                       R3              OBJECT ADDRESS
;22397  ;                       VA                ''        ''
;22398  ;                       R0              SOURCE LENGTH
;22399  ;                       D                 ''        ''
;22400  ;                       PC              SOURCE ADDRESS
;22401  ;                       R1                ''        ''
;22402  ;                       FLAG3           SET (MATCHC FLAG)
;22403  ;
;22404  ;       RESOURCES       TEMP6           HELPS IN DELTA PC SUB
;22405  ;
;22406  ;       SUBROUTINES     CS.R2.GETS.DELPC.R2   R2_DELTA PC'R2
;22407  ;*************************************************************************
```

```
                                ;22408          ;*************************************************************************
                                ;22409          ; CMPC3 and MOVC3 instruction flows begin at CS.OS.MOVC3.CMPC3.
                                ;22410          ; CMPC5, MOVC5, MOVTC, and MOVTUC enter at CS.OS.MOV.CMP.INT.
                                ;22411          ;*************************************************************************
                                ;22412
                                ;22413 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;22414
                                ;22415 CS.OS.MOVC3.CMPC3:
                                ;22416          ;-------------------------------;
                                ;22417          LOD INC BRA?,SIZE[WORD],        ;TEMP1_D_length, READ source address
                                ;22418          R[TEMP1]_D_ZEXT(M[MDR]),
U 03AE, 0085,259E,6190,4047,0012,0 ;22419       NEXT/OS.ADD
                                ;22420
                                ;22421 =0
                                ;22422 CS.OS.MOVC3.CMPC3.1:
                                ;22423          ;0------------------------------;
                                ;22424          PUSH,R[TEMP2]_M[MDR],           ;TEMP2_source address, READ dest.
U 0302, 0885,2592,41B0,8047,0412,0 ;22425       LOD INC BRA?,NEXT/OS.ADD        ;address
                                ;22426
                                ;22427          ;1------------------------------;
                                ;22428          M[TEMP6]_ZLIT8[1],SIZE[LONG],   ;TEMP6_100, anything pending?
U 0303, 0986,6DB7,0B20,0847,08A6,4 350* ;22429  ALLOW INT?
                                ;22430
                                ;22431 .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                                ;22432 =0
                                ;22433 CS.OS.MOV.CMP.INT:
                                ;22434          ;0------------------------------;
                                ;22435          R[R2]_0,SET FPD,                ;GPR2_0, FPD_0, nothing pending or
U 0A64, 0CEC,05B7,0034,8047,00A6,C ;22436       NEXT/CS.OS.MOV.CMP.IN           ;timer service
                                ;22437
                                ;22438          ;1------------------------------;
                                ;22439          PUSH,INTPEND OR TIMER?,         ;Yes, RETURN-1 or roll back
U 0A65, 0480,0036,4AF0,0047,04F7,0 ;22440       NEXT/IE.SERV.IP.TS2
                                ;22441
                                ;22442 =0
                                ;22443 CS.OS.MOV.CMP.IN:
                                ;22444          ;0------------------------------;
                                ;22445          PUSH,                           ;R2 GETS DELPC'(24 BITS)0
                                ;22446          R[R3]_VA_M[MDR],                ;R3,VA GET DESTINATION ADDRESS
U 0A6C, 0885,2592,4034,C4A7,04A8,E ;22447       NEXT/CS.R2.GETS.DELPC.R2        ;RET+1
                                ;22448
                                ;22449          ;1------------------------------;
U 0A6D, 0886,95B7,0030,0047,00A4,5 ;22450       M[TEMP9]_0                      ;Zero GPR0 decrement FLAG (used in
                                ;22451                                          ;pack routines)
                                ;22452
                                ;22453          ;-------------------------------;
U 0A45, 0886,C5B7,0030,0047,00A5,4 ;22454       M[FPDOFFSET]_0                  ;Guess a RETURN+4 from MM or IANDE
                                ;22455
                                ;22456          ;-------------------------------;
                                ;22457          PC_R[R1]_M[TEMP2],              ;GPR1_PC_Source address, which
U 0A54, 0484,2592,4674,4487,00A3,8 ;22458       IR<2-0>?                        ;inst. are we?
```

```
                                        ;22459 =000
                                        ;22460          ;000----------------------------;
                                        ;22461          WB_D-R[TEMP1],                  ;MOVC3[28], CC_src. len-src. len,
                                        ;22462          CCOP2,SIZE[LONG],               ;join up with other MOV's
U 0A38, 0080,0020,0020,5047,00A3,C      ;22463          NEXT/CS.M3
                                        ;22464
                                        ;22465 CS.C3:
                                        ;22466          ;001----------------------------;
                                        ;22467          R[R0]_D-0,CCOP1,SIZE[LONG],     ;CMPC3[29], GPR0_length,
U 0A39, 0C84,05B0,0A24,0847,08B5,2 386* ;22468          WX.EQ.0?,NEXT/CS.C3.1           ;set CC, is length =0?
                                        ;22469
                                        ;22470 =100
                                        ;22471 CS.M3:
                                        ;22472          ;100----------------------------;
                                        ;22473          R[R0]_Q_D,WX.NE.0?,             ;MOVC5[2C], load length, is it =0?
U 0A3C, 0484,05B2,5A74,0047,00AF,0      ;22474          NEXT/CS.M3.1                    ;(all MOV's join up here)
                                        ;22475
                                        ;22476          ;101----------------------------;
                                        ;22477          R[R0]_D-0,CCOP1,SIZE[LONG],     ;CMPC5[2D], GPR0_length,
U 0A3D, 0C84,05B0,0A24,0847,08B5,2 386* ;22478          WX.EQ.0?,NEXT/CS.C3.1           ;set CC, is length =0?
                                        ;22479
                                        ;22480          ;110----------------------------;
                                        ;22481          M[TEMP6]_ZLIT8[4],              ;MOVTC[2E], load code 400, join up
U 0A3E, 0586,6DB7,0030,2047,00A3,C      ;22482          NEXT/CS.M3                      ;with other MOV's
                                        ;22483
                                        ;22484          ;111----------------------------;
                                        ;22485          M[TEMP6]_ZLIT8[4],              ;MOVTUC[2F], ditto
U 0A3F, 0586,6DB7,0030,2047,00A3,C      ;22486          NEXT/CS.M3
```

```
                                ;22487 ;*************************************************************************
                                ;22488 ;       MOVC5 AND CMPC5 OPERAND SPECIFIER EVALUATION BEGINS HERE
                                ;22489 ;       MOVTC AND MOVTUC JOIN THE FLOW AT CS.OS.MOVTC.TUC.CONT
                                ;22490 ;*************************************************************************
                                ;22491
                                ;22492 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;22493
                                ;22494 CS.OS.MOVC5.CMPC5:
                                ;22495         ;-------------------------------;
                                ;22496         MDR_ZEXT(M[MDR]),               ;MDR source length for QREG placement,
                                ;22497         SIZE[WORD],LOD INC BRA?,        ;READ source address
U 03AF, 0481,2016,419D,8467,0012,0 ;22498       NEXT/OS.ADD
                                ;22499
                                ;22500 CS.OS.MOVC5.CMPC5.1:
                                ;22501         ;-------------------------------;
U 03B1, 0C86,A03A,403D,8047,00A4,0 ;22502       M[TEMP10]_Q                     ;MTMP10 GETS SRC LENGTH
                                ;22503
                                ;22504 .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                                ;22505 =000
                                ;22506
                                ;22507         ;000-----------------------------;
                                ;22508         PUSH,R[TEMP2]_M[MDR],           ;TEMP2_source address, READ fill char.
U 0A40, 0085,2592,41B0,8047,0410,0 ;22509       LOD INC BRA?,NEXT/OS.RED
                                ;22510
                                ;22511         ;001-----------------------------;
                                ;22512         PUSH,                           ;GET OBJECT LENGTH
                                ;22513         SIZE[BYTE],                     ;BYTESIZE
                                ;22514         R[TEMP5]_ZEXT(M[MDR]),          ;TEMP5 GETS FILL
                                ;22515         LOD INC BRA?,                   ;BUT INTO OS ROUTINE
U 0A41, 0485,259E,4181,4047,0410,0 ;22516       NEXT/OS.RED                     ;RET+1
                                ;22517
                                ;22518 ;*************************************************************************
                                ;22519 ;       THIS IS WHERE MOVTC MOVTUC JOIN THE FLOW
                                ;22520 ;*************************************************************************
                                ;22521 CS.OS.MOVTC.TUC.CONT:
                                ;22522         ;010-----------------------------;
                                ;22523         PUSH,                           ;GET OBJ ADDRESS
                                ;22524         SIZE[WORD],                     ;WORD'SWORTH
                                ;22525         R[TEMP3]_ZEXT(M[MDR]),          ;TEMP3 GETS OBJ LENGTH
                                ;22526         LOD INC BRA?,                   ;BUT INTO OS ROUTINE
U 0A42, 0885,259E,4190,C047,0412,0 ;22527       NEXT/OS.ADD                     ;RET+1
                                ;22528
                                ;22529         ;011-----------------------------;
                                ;22530         PUSH,                           ;EXPAND FILLER
                                ;22531         D_Q_M[TEMP10],                  ;D,Q GET SRC LENGTH
                                ;22532         SET_FLAG1,                      ;FLAG1 SET FOR MOVTC,TUC,C5,CMPC5
U 0A43, 0048,A592,7030,0047,04A9,A ;22533       NEXT/CS.EXP.FILLER              ;RET+1
                                ;22534
                                ;22535         ;100-----------------------------;
                                ;22536         M[TEMP10]_MB-R[TEMP3],CCOP1,    ;MTEMP10_src.len - dest.len, set CC,
U 0A44, 0086,A000,06D0,C847,08A6,1 338* ;22537  WB<31-30>?,SIZE[WORD]            ;what's the polarity of our result?
                                ;22538 =
```

```
                                        ;22539 =01
                                        ;22540          ;01-------------------------------;
U 0A61, 0080,3592,7030,0047,00A6,3      ;22541          D_Q_M[TEMP3]                    ;Source length >= Dest. length
                                        ;22542
                                        ;22543 =11
                                        ;22544          ;11-------------------------------;
                                        ;22545          M[TEMP6]_ZLIT8[1],SIZE[LONG],   ;(Src.len < dest. len.),
                                        ;22546          ALLOW INT?,                     ;load TEMP6, anything pending?
U 0A63, 0986,6DB7,0B20,0847,08A6,4 350* ;22547          NEXT/CS.OS.MOV.CMP.INT
```

```
                                ;22548          ;*********************************************************************
                                ;22549          ; Beginning of MOVTC and MOVTUC.
                                ;22550          ;*********************************************************************
                                ;22551
                                ;22552  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;22553
                                ;22554  CS.OS.MOVTC.MOVTUC:
                                ;22555          ;-------------------------------;
                                ;22556          MDR_ZEXT(M[MDR]),SIZE[WORD],    ;(Q_Source length), READ source
U 03B2, 0481,2016,419D,8467,0012,0 ;22557       LOD INC BRA?,NEXT/OS.ADD        ;address
                                ;22558
                                ;22559  CS.OS.MOVTC.MOVTUC.1:
                                ;22560          ;-------------------------------;
U 03B3, 0C86,A03A,403D,8047,00A6,8 ;22561       M[TEMP10]_Q                     ;MTMP10 GETS SRC LENGTH
                                ;22562
                                ;22563  .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                                ;22564  =00
                                ;22565          ;00-------------------------------;
                                ;22566          PUSH,                           ;GET FILL
                                ;22567          R[TEMP2]_M[MDR],                ;TEMP2 GETS SRC ADDRESS
                                ;22568          LOD INC BRA?,                   ;BUT INTO OS ROUTINE
U 0A68, 0085,2592,41B0,8047,0410,0 ;22569       NEXT/OS.RED                     ;RET+1
                                ;22570
                                ;22571          ;01-------------------------------;
                                ;22572          PUSH,                           ;GET TABLE ADDRESS
                                ;22573          SIZE[BYTE],                     ;BYTESIZE
                                ;22574          R[TEMP5]_ZEXT(M[MDR]),          ;TEMP5 GETS FILL
                                ;22575          LOD INC BRA?,                   ;BUT INTO OS ROUTINE
U 0A69, 0C85,259E,4181,4047,0412,0 ;22576       NEXT/OS.ADD                     ;RET+1
                                ;22577
                                ;22578          ;10-------------------------------;
                                ;22579          PUSH,                           ;GET OBJ LENGTH
                                ;22580          SET FLAG0,                      ;FLAG0 INDICATES MOVTC,TUC
                                ;22581          R[TEMP7]_M[MDR],                ;TEMP7 GETS TABLE ADDRESS
                                ;22582          LOD INC BRA?,                   ;BUT INTO OS ROUTINE
U 0A6A, 0045,2592,41B1,C047,0410,0 ;22583       NEXT/OS.RED                     ;RET+1
                                ;22584
                                ;22585          ;11-------------------------------;
U 0A6B, 0480,0036,4030,0047,00A4,2 ;22586       NEXT/CS.OS.MOVTC.TUC.CONT
                                ;22587  =
```

```
                                ;22588          ;*************************************************************************
                                ;22589          ; Beginning of SCANC and SPANC.
                                ;22590          ;*************************************************************************
                                ;22591
                                ;22592  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;22593
                                ;22594  CS.OS.SCANC.SPANC:
                                ;22595          ;-------------------------------;
                                ;22596          D_ZEXT(M[MDR]),SIZE[WORD],      ;DREG_length, READ address
U 03B4, 0081,2016,619D,8047,0012,0  ;22597      LOD INC BRA?,NEXT/OS.ADD
                                ;22598
                                ;22599  =00
                                ;22600  CS.OS.SCANC.SPANC.1:
                                ;22601          ;00-----------------------------;
                                ;22602          PUSH,                           ;GET TABLE ADDRESS
                                ;22603          R[TEMP1]_M[MDR],                ;R1 GETS ADDRESS
                                ;22604          LOD INC BRA?,                   ;BUT INTO OS ROUTINE
U 0274, 0885,2592,41B0,4047,0412,0  ;22605      NEXT/OS.ADD                     ;RET+1
                                ;22606
                                ;22607          ;01-----------------------------;
                                ;22608          PUSH,                           ;GET MASK
                                ;22609          R[TEMP2]_M[MDR],                ;TEMP2 GETS TABLE ADDRESS
                                ;22610          LOD INC BRA?,                   ;BUT INTO OS ROUTINE
U 0275, 0085,2592,41B0,8047,0410,0  ;22611      NEXT/OS.RED                     ;RET+1
                                ;22612
                                ;22613          ;10-----------------------------;
                                ;22614          SIZE[BYTE],                     ;BYTESIZE
U 0276, 0485,259E,4001,4047,00AC,0  ;22615      R[TEMP5]_ZEXT(M[MDR])           ;TEMP5 GETS MASK
                                ;22616  =
                                ;22617
                                ;22618  .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                                ;22619  =0
                                ;22620          ;0-----------------------------;
                                ;22621          PUSH,                           ;R3 GETS VA
                                ;22622          SET FPD,                        ;SET FIRST PART DONE
                                ;22623          R[R2]_M[PC],                    ;R2 GETS PC
U 0AC0, 08ED,A592,4034,8047,04AA,2  ;22624      NEXT/CS.OS.R3.VA.GET.TMP2       ;RET+1
                                ;22625
                                ;22626          ;1-----------------------------;
U 0AC1, 0884,1592,4034,4487,00A7,4  ;22627      PC_R[R1]_M[TEMP1]               ;PC GETS ADDRESS
                                ;22628
                                ;22629          ;-----------------------------;
                                ;22630          R[R0]_Q_D,                      ;R0,Q GETS SRC LENGTH
                                ;22631          CCOP2,                          ;SET CC
                                ;22632          SIZE[LONG],                     ;SO CCOP2 DOESN'T GET A VALIDITY CHECK
U 0A74, 0884,05B2,5A24,1047,08AD,8 322*  ;22633 WX.EQ.0?                        ;IF LENGTH =0 LEAVE
```

```
                                ;22634 =0
                                ;22635 CS.OS.SCSP.BEG:
                                ;22636          ;0------------------------------;
                                ;22637          M[FPDOFFSET]_ZLIT0[5],          ;Load FPDOFFSET, I/O FLAG_0
U 0AD8, 0116,CC37,0030,2847,00B6,B ;22638       NEXT/CS.SCSP,CLEAR FLAG2
                                ;22639
                                ;22640 CS.SCSP.EXIT:
                                ;22641          ;1------------------------------;
                                ;22642          PC_R[R2],CLEAR FPD,             ;Restore PC, FPD_0, IRD1 in MOVTC code
U 0AD9, 04E0,05BE,4034,8487,00B1,B ;22643       NEXT/CS.MTC.R2_0.IRD1
```

```
                                ;22644          ;*********************************************************************
                                ;22645          ; Beginning of LOCC and SKPC.
                                ;22646          ;*********************************************************************
                                ;22647
                                ;22648 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;22649
                                ;22650 CS.OS.LOCC.SKPC:
                                ;22651          ;-------------------------------;
                                ;22652          R[TEMP5]_ZEXT(M[MDR]),          ;TEMP5_character, READ length
                                ;22653          SIZE[BYTE],LOD INC BRA?,
U 03B5, 0C85,259E,4181,4047,0010,0  ;22654      NEXT/OS.RED
                                ;22655
                                ;22656 =0
                                ;22657 CS.OS.LOCC.SKPC.1:
                                ;22658          ;0-------------------------------;
                                ;22659          PUSH,                           ;GET ADDRESS
                                ;22660          SIZE[WORD],                     ;WORD'SWORTH
                                ;22661          D_ZEXT(M[MDR]),                 ;R0,D,GETS LENGTH
                                ;22662          LOD INC BRA?,                   ;BUT INTO OS ROUTINE
U 030A, 0881,2016,619D,8047,0412,0  ;22663      NEXT/OS.ADD                     ;RET+1
                                ;22664
                                ;22665          ;1-------------------------------;
U 030B, 08E9,A592,5030,0047,004E,4  ;22666      SET FPD,Q_M[PC]                 ;FPD_1, QREG_PC
                                ;22667
                                ;22668 .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                                ;22669
                                ;22670 =0
                                ;22671          ;0-------------------------------;
                                ;22672          PUSH,                           ;EXPAND FILLER
                                ;22673          PC_R[R1]_M[MDR],                ;R1,PC GETS ADDRESS
U 0AE4, 0485,2592,4034,4487,04A9,A  ;22674      NEXT/CS.EXP.FILLER              ;RET+1
                                ;22675
                                ;22676          ;1-------------------------------;
U 0AE5, 0D86,CC37,0670,3047,00A4,6  ;22677      M[FPDOFFSET]_ZLIT0[6],IR<2-0>?   ;Load FPDOFFSET
                                ;22678
                                ;22679
                                ;22680 =110
                                ;22681 CS.OS.LOC:
                                ;22682          ;110-----------------------------;
                                ;22683          R[R0]_D-0,CCOP1,SIZE[WORD],     ;LOCC[3A], set CC, is length = 0?
U 0A46, 0C84,05B0,0A14,0847,08AF,6 386*  ;22684 WX.EQ.0?,NEXT/CS.LOC
                                ;22685
                                ;22686          ;111-----------------------------;
                                ;22687          R[R0]_D-0,CCOP1,SIZE[WORD],     ;GPR0_length, set CC, is length =0?
U 0A47, 0C84,05B0,0A14,0847,08AF,C 386*  ;22688 WX.EQ.0?,NEXT/CS.SK               ;(into SKPC flows)
```

```
                                ;22689          ;*********************************************************************************
                                ;22690          ; Beginning of MATCHC.
                                ;22691          ;*********************************************************************************
                                ;22692
                                ;22693  .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;22694
                                ;22695  CS.OS.MATCHC:
                                ;22696          ;-------------------------------;
                                ;22697          R[TEMP1]_ZEXT(M[MDR]),          ;TEMP1_low WORD from MDR, READ obj.
                                ;22698          SIZE[WORD],LOD INC BRA?,        ;address
U 03B6, 0485,259E,4190,4047,0012,0  ;22699      NEXT/OS.ADD
                                ;22700
                                ;22701  =000
                                ;22702  CS.OS.MATCHC.1:
                                ;22703          ;000----------------------------;
                                ;22704          PUSH,R[TEMP2]_M[MDR],           ;TEMP2_obj. address, read src. length
U 0320, 0085,2592,41B0,8047,0410,0  ;22705      LOD INC BRA?,NEXT/OS.RED
                                ;22706
                                ;22707          ;001----------------------------;
                                ;22708          PUSH,                           ;GET SRC ADDRESS
                                ;22709          SIZE[WORD],                     ;WORD'SWORTH
                                ;22710          D_ZEXT(M[MDR]),                 ;D GET SRC LENGTH
                                ;22711          LOD INC BRA?,                   ;BUT INTO OS ROUTINE
U 0321, 0881,2016,619D,8047,0412,0  ;22712      NEXT/OS.ADD                     ;RET+1
                                ;22713
                                ;22714          ;010----------------------------;
                                ;22715          PUSH,R[R2]_Q_M[TEMP1],          ;GPR2 & QREG_TEMP1, set CC, FPD_1,
                                ;22716          CCOP1,SIZE[WORD],SET FPD,       ;GPR2_deltaPC in BYTE #3
U 0322, 04EC,1592,5014,8847,04A8,E  ;22717      NEXT/CS.R2.GETS.DELPC.R2
                                ;22718
                                ;22719          ;011----------------------------;
                                ;22720          PUSH,SET FLAG3,                 ;MATCHC FLAG_1, GPR3_VA,
                                ;22721          PC_R[R1]_M[MDR],                ;R1,PC GET SOURCE ADDRESS
U 0323, 0C5D,2592,4034,4487,04AA,2  ;22722      NEXT/CS.OS.R3.VA.GET.TMP2       ;RET+1
                                ;22723
                                ;22724          ;100----------------------------;
U 0324, 0484,05B2,4034,0047,003E,B  ;22725      R[R0]_D                         ; R0 GETS D (SRC LENGTH)
                                ;22726  =
                                ;22727
                                ;22728  3EB:
                                ;22729          ;******FORCE ADDRESS************;
                                ;22730          WB_Q,WX.EQ.0?,                  ; Q TO WBUS
U 03EB, 0480,003A,4A3D,8047,00B7,A  ;22731      NEXT/CS.MAT.SET.FPDOFF          ; IS OBJECT LENGTH = 0?
```

```
                                ;22732 .TOC     "       Character String                : General Routines"
                                ;22733
                                ;22734 .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                                ;22735
                                ;22736 ;*************************************************************************
                                ;22737 ;               OUTPUT                  R2              DELTA PC'0'R2(8)(8)(16)
                                ;22738 ;               RESOURCES               PC              FOR DELTA CALCULATION
                                ;22739 ;                                       PCBACK          FOR DELTA CALCULATION
                                ;22740 ;                                       TEMP4           FOR DELTA CALCULATION
                                ;22741 ;*************************************************************************
                                ;22742
                                ;22743 CS.R2.GETS.DELPC.R2:
                                ;22744         ;-------------------------------;
U 0A8E, 0485,A592,4031,0047,00A9,6  ;22745         R[TEMP4]_M[PC]                  ;TEMP4 GETS PC
                                ;22746
                                ;22747         ;-------------------------------;
U 0A96, 0C85,9003,0031,0047,00A9,8  ;22748         R[TEMP4]_RB-M[PCBACK]           ;TEMP4 GETS PC-PCBACK
                                ;22749
                                ;22750         ;-------------------------------;
                                ;22751         R[R2]_RB.OR.(M[TEMP4].RL.24),   ;R2_DELPC'R2(DELPC IS 8 BITS)
U 0A98, 0884,43BE,4084,8047,0000,1  ;22752         RETURN [1]                      ;RETURN +1
                                ;22753
                                ;22754
                                ;22755 ;*************************************************************************
                                ;22756 ;               OUTPUT                  TEMP5           TEMP5'TEMP5'TEMP5'TEMP5
                                ;22757 ;*************************************************************************
                                ;22758
                                ;22759 CS.EXP.FILLER:
                                ;22760         ;-------------------------------;
U 0A9A, 0486,5392,4020,0047,00AA,0  ;22761         M[TEMP5]_MB.OR.(MB.RL.8)        ;TEMP5_0'0'TEMP5'TEMP5(TEMP5 IS 8 BITS)
                                ;22762
                                ;22763         ;-------------------------------;
                                ;22764         M[TEMP5]_MB.OR.(MB.RL.16),      ;TEMP5_TEMP5'TEMP5(TEMP5 IS 16 BITS)
U 0AA0, 0886,5392,4090,0047,0000,1  ;22765         RETURN [T]                      ;RET+1
                                ;22766
                                ;22767
                                ;22768 ;*************************************************************************
                                ;22769
                                ;22770 CS.OS.R3.VA.GET.TMP2:
                                ;22771         ;-------------------------------;
                                ;22772         VA_R[R3]_M[TEMP2],              ;R3 VA GETS TMP2
U 0AA2, 0884,2592,40B4,C4A7,0000,1  ;22773         RETURN [T]                      ;RET+1
```

```
                                ;22774 ;*************************************************************************
                                ;22775 ;               INPUT                   R2              DELTA PC IN <31-24>
                                ;22776 ;               OUTPUT                  PC              ORIGINAL VALUE
                                ;22777 ;               RESOURCES               TEMP0           FOR RESTORATION
                                ;22778 ;*************************************************************************
                                ;22779 CS.GEN.RESPC:
                                ;22780         ;-------------------------------;
U 0AA5, 0086,02B7,0024,8047,00AA,A ;22781      M[TEMP0]_R[R2].RR.24            ;TEMP0_DELTA PC IN LOWEST 8 BITS
                                ;22782
                                ;22783 CS.GEN.RESPC.1:
                                ;22784         ;-------------------------------;
U 0AAA, 0D86,0C12,0031,F847,00AB,0 ;22785      M[TEMP0]_MB.AND.ZLIT0[03F]      ;TEMP0_TEMP0 AND 3F (DELTA PC ONLY)
                                ;22786
                                ;22787         ;-------------------------------;
                                ;22788         PC_M[PCBACK]+R[TEMP0],          ;PC_PCBACK+DELTA PC
U 0AB0, 0481,9001,00B0,0487,0000,1 ;22789      RETURN [1]                      ;RET+1
                                ;22790
                                ;22791
                                ;22792 ;*************************************************************************
                                ;22793 ;               INPUT                   FPDOFFSET       HAS 80000000 OR 40000000
                                ;22794 ;                                       MTEMP8          3FFFFFFF
                                ;22795 ;                                       R2              GOOD<29-0>
                                ;22796 ;               OUTPUT                  R2              BIT<29-0>SAME AS INPUT
                                ;22797 ;                                                       BIT<31-30>_FPDOFF<31-30>
                                ;22798 ;*************************************************************************
                                ;22799 CS.FIX.MATCHC.IND:
                                ;22800         ;-------------------------------;
U 0AB2, 0484,8002,0034,8047,00AB,4 ;22801      R[R2]_M[TEMP8].AND.RB           ;R2_R2 STRIPPED OF BITS 31 AND 30
                                ;22802
                                ;22803         ;-------------------------------;
                                ;22804         R[R2]_M[FPDOFFSET].OR.RB,       ;PUT BITS 31-30 OF FPDOFFSET INTO R2
U 0AB4, 0884,C002,40B4,8047,0000,4 ;22805      RETURN [4]                      ;RET+4
```

```
;22806 .TOC     "       Character String                : MOVC3"
;22807
;22808          ;*****************************************************************************
;22809          ;       MOVC3
;22810          ;
;22811          ;       INPUT           R0              LENGTH
;22812          ;                       D                 "
;22813          ;                       Q                 "
;22814          ;                       R1              SOURCE ADDRESS
;22815          ;                       PC                "
;22816          ;                       TEMP2             "       "
;22817          ;                       R3              DESTINATION ADDRESS
;22818          ;                       VA                "
;22819          ;                       R2              DELTA PC'0(8BITS'24BITS)
;22820          ;                       TEMP6           1(ROTATED LEFT BY 8)
;22821          ;                       FPDOFFSET       0 for RETURN+4
;22822          ;                       CC SET          (R0-R0)
;22823          ;                       FPD             Set
;22824          ;
;22825          ;       RESOURCES       R0              LENGTH
;22826          ;                       R1              SOURCE ADDRESS
;22827          ;                       R2              DELTA PC'0(8BITS'24BITS)
;22828          ;                       R3              DESTINATION ADDRESS
;22829          ;                       D               LENGTH
;22830          ;                       Q                 "
;22831          ;                       TEMP0<1-0>      Low 2 bits of length.  After
;22832          ;                                       moving the LONGWORDs of an
;22833          ;                                       aligned dest., these bits are
;22834          ;                                       the number of extra BYTEs to
;22835          ;                                       move.
;22836          ;                       TEMP2
;22837          ;                       TEMP4
;22838          ;                       TEMP6           CODE
;22839          ;                       RTEMP12         Holds XB data
;22840          ;                       MTEMP9          FPD GPR0 decrement flag.
;22841          ;                                       =0 if GPR0 is decremented after
;22842          ;                                       a READ/WRITE. (BYTE move)
;22843          ;                                       >0 if GPR0 is decremented before
;22844          ;                                       a READ/WRITE and must be
;22845          ;                                       adjusted before packing.
;22846          ;                       MTMP11          1
;22847          ;                       PC              SOURCE ADDRESS
;22848          ;                       VA              DESTINATION ADDRESS
;22849          ;                       CC SET          (R0-R0)
;22850          ;                       FLAG0           0
;22851          ;                       FLAG1           0
;22852          ;                       FLAG3           0
```

```
                    ;22853          ;       OUTPUT          R0              0
                    ;22854          ;                       R1              ADD OF ONE BYTE BEYOND
                    ;22855          ;                                       THE SOURCE STRING
                    ;22856          ;                       R2              0
                    ;22857          ;                       R3              ADD OF ONE BYTE BEYOND
                    ;22858          ;                                       THE DESTINATION STRING
                    ;22859          ;                       R4              0
                    ;22860          ;                       R5              0
                    ;22861          ;                       CC              NZVC=0100 (Z SET)
                    ;22862          ;
                    ;22863          ;       SUBROUTINES     CS.GEN.RESPC
                    ;22864          ;*******************************************************************************
```

```
;22865  ;*****************************************************************************
;22866  ;       MOVC5
;22867  ;
;22868  ;       INPUT           R0              MIN(SRC LGTH,DEST LGTH)
;22869  ;                       Q                  ''   ''          ''
;22870  ;                       D
;22871  ;                       R1              SOURCE ADDRESS
;22872  ;                       PC                 ''      ''
;22873  ;                       TEMP2
;22874  ;                       TEMP5           FILL(EXPANDED)
;22875  ;                       TEMP3           DESTINATION LENGTH
;22876  ;                       FLAG1           SET(FOR CMPC5 AND MOVC5)
;22877  ;                       MTEMP10         SRC LGTH - DST LGTH
;22878  ;                       FPDOFFSET       0 for RETURN+4
;22879  ;                                       ZLITO[9.] in MOVC5 seperate
;22880  ;                                       code for pack routine.
;22881  ;                       R3              DESTINATION ADDRESS
;22882  ;                       VA                 ''          ''
;22883  ;                       R2              DELTA PC'0(8BITS'24BITS)
;22884  ;                       TEMP6           1(ROTATED LEFT BY 8)
;22885  ;                       CC SET          (TEMP10_R0-TMP3)
;22886  ;                       FPD             SET
;22887  ;
;22888  ;       RESOURCES       SAME AS MOVC3 WITH THE FOLLOWING
;22889  ;                       TEMP5           FILL CHARACTER
;22890  ;                       MTMP10          DIFF OF SRC & DEST LGTH
;22891  ;                       FLAG1           1
;22892  ;
;22893  ;       OUTPUT          R0              # OF UNMOVED BYTES IN
;22894  ;                                       SRC STR
;22895  ;                                       (NE.0 IF SRC>DEST LGTH)
;22896  ;                       R1              ADD OF ONE BYTE BEYOND
;22897  ;                                       THE LAST BYTE IN SRC
;22898  ;                                       STRING THST WAS MOVED
;22899  ;                       R2              0
;22900  ;                       R3              ADD OF ONE BYTE BEYOND
;22901  ;                                       THE DESTINATION STRING
;22902  ;                       R4              0
;22903  ;                       R5              0
;22904  ;                       CC              N_SRCLEN LSS DSTLEN
;22905  ;                                       Z_SRCLEN EQL DESLEN
;22906  ;                                       V_0
;22907  ;                                       C_SRCLEN LSSU DSTLEN
;22908  ;*****************************************************************************
```

```
                                ;22909 =0
                                ;22910 CS.M3.1:
                                ;22911          ;0--------------------------------;
                                ;22912          IR<2-0>?,                       ;Length =0, which inst. are we?
U 0AF0, 0080,0036,4670,0047,00A5,6 ;22913       NEXT/CS.M3.MTUC
                                ;22914
                                ;22915          ;1--------------------------------;
                                ;22916          WB_M[TEMP2]-R[R3],              ;TEMP2 < R3?
U 0AF1, 0C80,2000,0024,C047,00AB,6 ;22917       SIZE[LONG]                      ;CHECK
                                ;22918
                                ;22919          ;--------------------------------;
                                ;22920          ALKC?,                          ;IS TEMP2 < R3???
U 0AB6, 0480,0037,C6F0,0047,00A3,A ;22921       NEXT/CS.SRCADD.VS.DESTADD       ;
                                ;22922
                                ;22923
                                ;22924 =00
                                ;22925 =01
                                ;22926 CS.MOV.R4.GETS.0:
                                ;22927          ;01-------------------------------;
                                ;22928          PUSH,                           ;RESTORE PC
                                ;22929          R[R4]_0,                        ;R4 GETS 0
U 0AAD, 0C84,05B7,0035,0047,04AA,5 ;22930       NEXT/CS.GEN.RESPC               ;RET+1
                                ;22931
                                ;22932          ;10-------------------------------;
                                ;22933          R[R2]_0,                        ;R2 GETS 0
                                ;22934          CLEAR_FPD,                      ;CLEAR FPD
                                ;22935          FLAG0?,                         ;MOVTC,MOVTUC?
U 0AAE, 08E4,05B7,0434,8047,00AF,8 ;22936       NEXT/CS.MOV.R5.GETS.0           ;
                                ;22937
                                ;22938          ;11-------------------------------;
                                ;22939          WB_M[TEMP10],WB<31-30>?,         ;MOVC5 flow, is MTEMP10 < 0?
U 0AAF, 0080,A592,46F0,0047,00AB,1 ;22940       NEXT/CS.M5
                                ;22941
                                ;22942
                                ;22943 =0
                                ;22944 CS.MOV.R5.GETS.0:
                                ;22945          ;0--------------------------------;
U 0AF8, 0484,05B7,0135,4047,003F,9 ;22946       R[R5]_0,IRD1                    ;GPR5_0, **IRD1**
                                ;22947
                                ;22948          ;1--------------------------------;
U 0AF9, 0480,05BF,6034,C047,00AB,B ;22949       D_R[R3]                         ;GPR5_GPR3
                                ;22950
                                ;22951          ;--------------------------------;
U 0ABB, 0C84,05B2,4035,4047,00AB,F ;22952       R[R5]_D
                                ;22953
                                ;22954          ;--------------------------------;
U 0ABF, 0084,7592,4134,C047,003F,9 ;22955       R[R3]_M[TEMP7],IRD1             ;GPR3_table address, **IRD1**
```

```
                                ;22956          ;*************************************************************************
                                ;22957          ; Here to diverge flow if overlapping MOVE.  TEMP2 contains source
                                ;22958          ; address and GPR3 has the destination address.  If TEMP2 < GPR3 and
                                ;22959          ; TEMP2+(src.len) > GPR3 we have an overlapping MOVE.
                                ;22960          ;*************************************************************************
                                ;22961 =10
                                ;22962 CS.SRCADD,VS.DESTADD:
                                ;22963          ;10-------------------------------;
                                ;22964          FLAG0?,                         ;TMP2> = R3 MOVTC(TUC)?
U 0A3A, 0880,0036,4430,0047,00B3,C ;22965       NEXT/CS.MOV.VA.END.IN.00        ;NO OVERLAP
                                ;22966
                                ;22967 =11
                                ;22968          ;11-------------------------------;
U 0A3B, 0886,8021,0030,8047,00AC,4 ;22969       M[TEMP8]_D+R[TEMP2]             ;MTMP8_D+TMP2 MOV BACKWARD
                                ;22970
                                ;22971          ;---------------------------------;
                                ;22972          WB_R[R3]-M[TEMP8],              ;IS DEST ADD > LARGEST SRC ADD WRITEN
U 0AC4, 0480,8003,0024,C047,00AC,6 ;22973       SIZE[LONG]                      ;CHECK
                                ;22974
                                ;22975          ;---------------------------------;
U 0AC6, 0C80,0037,C6F0,0047,00A5,2 ;22976       ALKC?                           ;IS DEST ADD>LARGEST SRC ADD?
                                ;22977
                                ;22978 =10
                                ;22979          ;10-------------------------------;
                                ;22980          FLAG0?,                         ;NO OVERLAP MOVTC(TUC)?
U 0A52, 0880,0036,4430,0047,00B3,C ;22981       NEXT/CS.MOV.VA.END.IN.00
                                ;22982
                                ;22983 =11
                                ;22984          ;11-------------------------------;
U 0A53, 0884,8592,4034,4047,00AC,B ;22985       R[R1]_M[TEMP8]                  ;YES OVERLAP,R1_FINAL SRC ADDRESS
                                ;22986
                                ;22987          ;---------------------------------;
U 0ACB, 0084,0039,0034,C047,00AC,C ;22988       R[R3]_RB+Q                      ;FINAL OBJ ADDRESS
                                ;22989
                                ;22990          ;---------------------------------;
U 0ACC, 0C86,4E7D,0034,C4A7,00AC,E ;22991       M[TEMP4]_VA_R[R3]-1             ;TMP4,VA_FURTHEST ADD FOR WRITING BACK
                                ;22992
                                ;22993          ;---------------------------------;
U 0ACE, 0880,8080,243D,8487,00B3,8 ;22994       PC_D_M[TEMP8]-1,FLAG0?          ;PC_furthest address for backwards READ,
                                ;22995                                          ;are we MOVTC(TUC)?
                                ;22996
                                ;22997 =0
                                ;22998          ;0-------------------------------;
                                ;22999          M[TEMP6]_ZLIT8[2],              ;No, load FPD code 200, anything
                                ;23000          MM.ALLOW.INT?,                  ;pending?
U 0B38, 0D86,6DB7,02F0,1047,00A4,8 ;23001       NEXT/CS.M3.WR.BACK
                                ;23002
                                ;23003          ;1--------------------------------;
U 0B39, 0586,CC37,0030,4847,00AD,0 ;23004       M[FPDOFFSET]_ZLIT0[9]           ;Yes, load FPDOFFSET
                                ;23005
                                ;23006          ;---------------------------------;
                                ;23007          R[TEMP13]_M[PC],                ;Move backwards in MOVTUC code
U 0AD0, 0C85,A592,4033,4047,00B1,2 ;23008       NEXT/CS.MOVTC.TUC.BACK
```

```
                                ;23009          ;*************************************************************************
                                ;23010          ; This is the backwards READ/WRITE loop for writing backwards in a
                                ;23011          ; MOVC3 or MOVC5.  =000 is nothing pending or timer service, =010 is
                                ;23012          ; something pending, RETURN-1 or pack up.
                                ;23013          ;*************************************************************************
                                ;23014 =000
                                ;23015 CS.M3.WR.BACK:
                                ;23016          ;000-------------------------------;
                                ;23017          R[TEMP12]_XB PC_PC+1,            ;Move a BYTE
U 0A48, 0C85,7592,4003,2047,00A4,B ;23018       NEXT/CS.M3.WR.BACK.1
                                ;23019
                                ;23020          ;001-------------------------------;
                                ;23021          R[TEMP12]_XB PC_PC+1,            ;Move a BYTE
U 0A49, 0C85,7592,4003,2047,00A4,B ;23022       NEXT/CS.M3.WR.BACK.1
                                ;23023
                                ;23024          ;010-------------------------------;
                                ;23025          PUSH,INTPEND OR TIMER?,
U 0A4A, 0480,0036,4AF0,0047,04F7,0 ;23026       NEXT/IE.SERV.IP.TS2
                                ;23027
                                ;23028 =100
                                ;23029          ;100-------------------------------;
                                ;23030          M[TEMP5]_MB.AND.ZLIT16[OFF],     ;BUS exception, pack up
U 0A4C, 0986,5D12,0037,F847,00BA,6 ;23031       NEXT/CS.MOV.PACK.BEGIN
                                ;23032
                                ;23033          ;101-------------------------------;
                                ;23034          M[TEMP5]_MB.AND.ZLIT16[OFF],     ;BUS exception, pack up
U 0A4D, 0986,5D12,0037,F847,00BA,6 ;23035       NEXT/CS.MOV.PACK.BEGIN
                                ;23036
                                ;23037          ;110-------------------------------;
                                ;23038          M[TEMP5]_MB.AND.ZLIT16[OFF],     ;Ints. pending, pack up
U 0A4E, 0986,5D12,0037,F847,00BA,6 ;23039       NEXT/CS.MOV.PACK.BEGIN
                                ;23040 =
                                ;23041
                                ;23042 =0**
                                ;23043 CS.M3.WR.BACK.1:
                                ;23044          ;0**-------------------------------;
                                ;23045          WRITE R[TEMP12],SIZE[BYTE],
U 0A4B, 0480,05BE,4003,05D8,00AD,2 ;23046       NEXT/CS.M3.WR.BACK.1A
                                ;23047
                                ;23048          ;1**-------------------------------;
                                ;23049          PC_M[PC]-ZLIT0[1],               ;BUS exception, fix PC, pack up
U 0A4F, 0581,AC10,0030,0C87,00FC,9 ;23050       NEXT/CS.MOV.PACK                 ;in IANDE code
```

```
                                        ;23051 CS.M3.WR.BACK.1A:
                                        ;23052          ;--------------------------------;
U 0AD2, 0C80,00A0,203D,8487,00AD,4      ;23053          PC_D_D-1                        ;Dec. PC by 1
                                        ;23054
                                        ;23055          ;--------------------------------;
U 0AD4, 0484,0E7D,0A34,0047,08B3,A 378* ;23056          R[R0]_RB-1,WX.EQ.0?             ;Dec. length, are we finished?
                                        ;23057
                                        ;23058 =0
                                        ;23059          ;0-------------------------------;
                                        ;23060          VA_M[TEMP4]_MB-1,MM.ALLOW.INT?, ;No, dec. dest. address, anything
U 0B3A, 0486,4080,02FD,84A7,00A4,8      ;23061          NEXT/CS.M3.WR.BACK              ;pending?
                                        ;23062
                                        ;23063          ;1-------------------------------;
                                        ;23064          VA_M[TEMP4]_MB-1,FLAG1?,        ;Yes, are we MOVC5?
U 0B3B, 0886,4080,05FD,84A7,00AA,D      ;23065          NEXT/CS.MOV.R4.GETS.0
```

```
                                ;23066          ;*****************************************************************************
                                ;23067          ; The loop for moving characters (normal move) is here.
                                ;23068          ; =0 is MOVC3 or MOVC5.  =1 is MOVTC or MOVTUC.
                                ;23069          ;*****************************************************************************
                                ;23070 =0
                                ;23071 CS.MOV.VA.END.IN.00:
                                ;23072          ;0------------------------------;
                                ;23073          WB_R[R3],                       ;Are we LONGWORD aligned?
                                ;23074          WB<1-0>.NE.0?,
U 0B3C, 0480,05BE,43B4,C047,00A5,0  ;23075      NEXT/CS.M3.WR
                                ;23076
                                ;23077          ;1------------------------------;
                                ;23078          M[FPDOFFSET]_ZLIT0[9],           ;Is anything pending?
U 0B3D, 0186,CC37,02F0,4847,00AB,C  ;23079      MM.ALLOW.INT?,NEXT/CS.MTC.WR
                                ;23080
                                ;23081
                                ;23082 =000
                                ;23083 CS.M3.WR:
                                ;23084          ;000-----------------------------;
                                ;23085          WB_R[R0]-CONX(4),WB<31-30>?,     ;We are LONGWORD aligned, > 1 LONGWORD
U 0A50, 0C80,073C,06E4,0047,08A9,5 376*  ;23086  NEXT/CS.M3.WR.L                  ;left?
                                ;23087
                                ;23088 CS.M3.WR.B:
                                ;23089          ;001-----------------------------;
                                ;23090          R[TEMP12]_XB_PC_PC+1,            ;Not LONGWORD aligned, move BYTEs
U 0A51, 0485,7592,4003,2047,00A5,B  ;23091      NEXT/CS.M3.WR.B.A                ;until done or aligned
                                ;23092
                                ;23093 =101
                                ;23094          ;101-----------------------------;
                                ;23095          M[TEMP5]_MB.AND.ZLIT16[OFF],     ;BUS exception, pack up
U 0A55, 0986,5D12,0037,F847,00BA,6  ;23096      NEXT/CS.MOV.PACK.BEGIN
                                ;23097 =
                                ;23098
                                ;23099 =0**
                                ;23100 CS.M3.WR.B.A:
                                ;23101          ;0**-----------------------------;
                                ;23102          WRITE R[TEMP12],SIZE[BYTE],
U 0A5B, 0C80,05BE,4003,05D8,00AD,C  ;23103      NEXT/CS.M3.WR.B.B
                                ;23104
                                ;23105          ;1**-----------------------------;
                                ;23106          PC_M[PC]-ZLIT0[1],               ;BUS exception, fix PC, pack up
U 0A5F, 0581,AC10,0030,0C87,00FC,9  ;23107      NEXT/CS.MOV.PACK                 ;in IANDE code
```

```
                                        ;23108 CS.M3.WR.B.B:
                                        ;23109          ;--------------------------------;
                                        ;23110          R[R0]_D_RB-1,                   ;Dec. length, are we done?
U 0ADC, 0884,0E7D,2A34,0047,08B3,E 378* ;23111          WX.EQ.0?
                                        ;23112
                                        ;23113 =0
                                        ;23114          ;0--------------------------------;
                                        ;23115          VA_R[R3]_RB+1,WB<1-0>.NE.0?,    ;Inc. dest. address, are we aligned?
U 0B3E, 0C84,0E7C,03B4,C4A7,08A5,0 374* ;23116          NEXT/CS.M3.WR
                                        ;23117
                                        ;23118          ;1--------------------------------;
                                        ;23119          VA_R[R3]_RB+1,                  ;Yes, inc. dest. add., finish inst.
U 0B3F, 0084,0E7C,0034,C4A7,00B4,0      ;23120          NEXT/CS.M3.R1_PC
```

```
                                ;23121          ;*****************************************************************************
                                ;23122          ; Writing forward, LONGWORD aligned.  =01 is >=1 LONGWORD left to
                                ;23123          ; move, =11 is < 1 LONGWORD left, READ/WRITE BYTEs.
                                ;23124          ;*****************************************************************************
                                ;23125 =01
                                ;23126 CS.M3.WR.L:
                                ;23127          ;01-----------------------------;
                                ;23128          M[TEMP9]_ZLIT0[3],              ;Set FPD GPR0 decrement flag
U 0A95, 0D86,9C37,0030,1847,00AD,E ;23129       NEXT/CS.M3.WR.L1
                                ;23130
                                ;23131 CS.M3.WR.LB:
                                ;23132          ;11-----------------------------;
U 0A97, 0886,95B7,0030,0047,00A5,1 ;23133       M[TEMP9]_0,NEXT/CS.M3.WR.B      ;Clear FPD GPR0 decrement flag
                                ;23134
                                ;23135 CS.M3.WR.L1:
                                ;23136          ;-------------------------------;
U 0ADE, 0886,05BE,4034,0047,00AE,0 ;23137       M[TEMP0]_R[R0]                  ;<1-0> hold # of extra BYTEs to move
                                ;23138                                          ;after finishing LONGWORDs
                                ;23139
                                ;23140          ;-------------------------------;
U 0AE0, 0084,9003,C034,0047,00AE,8 ;23141       R[R0]_NOT(M[TEMP9]).AND.RB      ;GPR0 becomes evenly divisible by 4
                                ;23142
                                ;23143          ;-------------------------------;
                                ;23144          R[R0]_RB-CONX(4),ALUS_UNSGN,    ;Dec. length BEFORE write,
U 0AE8, 0084,073C,02E4,10C7,00A5,8 ;23145       MM.ALLOW.INT?                   ;ALUS_polarity, anything pending?
```

```
                                ;23146          ;*************************************************************************
                                ;23147          ; =000 is nothing pending, =010 is ints. or timer pending.
                                ;23148          ; IE.SERV.IP.TS2 will RETURN -1 if timer service, so =001 is a duplicate
                                ;23149          ; of =000.
                                ;23150          ;*************************************************************************
                                ;23151 =000
                                ;23152 CS.M3.WR.L.LOP:
                                ;23153          ;000--------------------------;
                                ;23154          R[TEMP12] XB PC_PC+4,           ;Move a LONGWORD
U 0A58, 0885,7592,4023,2047,00A7,8  ;23155      NEXT/CS.M3.WR.L.LOP1
                                ;23156
                                ;23157          ;001--------------------------;
                                ;23158          R[TEMP12] XB PC_PC+4,           ;Move a LONGWORD
U 0A59, 0885,7592,4023,2047,00A7,8  ;23159      NEXT/CS.M3.WR.L.LOP1
                                ;23160
                                ;23161          ;010--------------------------;
                                ;23162          PUSH.INTPEND OR TIMER?,         ;Something pending, RETURN -1 to
U 0A5A, 0480,0036,4AF0,0047,04F7,0  ;23163      NEXT/IE.SERV.IP.TS2             ;loop or pack up
                                ;23164
                                ;23165 =100
                                ;23166          ;100--------------------------;
                                ;23167          M[TEMP5] MB.AND.ZLIT16[OFF],    ;BUS exception, pack up
U 0A5C, 0986,5D12,0037,F847,00BA,6  ;23168      NEXT/CS.MOV.PACK.BEGIN
                                ;23169
                                ;23170          ;101--------------------------;
                                ;23171          M[TEMP5] MB.AND.ZLIT16[OFF],    ;BUS exception, pack up
U 0A5D, 0986,5D12,0037,F847,00BA,6  ;23172      NEXT/CS.MOV.PACK.BEGIN
                                ;23173
                                ;23174          ;110--------------------------;
                                ;23175          M[TEMP5] MB.AND.ZLIT16[OFF],    ;Ints. pending, pack up
U 0A5E, 0986,5D12,0037,F847,00BA,6  ;23176      NEXT/CS.MOV.PACK.BEGIN
                                ;23177 =
                                ;23178
                                ;23179 =0**
                                ;23180 CS.M3.WR.L.LOP1:
                                ;23181          ;0**-------------------------;
                                ;23182          WRITE R[TEMP12],SIZE[LONG],
U 0A78, 0880,05BE,4023,05D8,00AE,A  ;23183      NEXT/CS.M3.WR.L.LOP1.A
                                ;23184
                                ;23185          ;1**--------------------------;
                                ;23186          PC_M[PC]-ZLIT0[4],              ;BUS exception, fix PC, pack up
U 0A7C, 0981,AC10,0030,2487,00FC,9  ;23187      NEXT/CS.MOV.PACK
```

```
                        ;23188 CS.M3.WR.L.LOP1.A:
                        ;23189          ;--------------------------------;
                        ;23190          R[R0]_D_RB-CONX(4),              ;Dec. length for next pass thru loop,
U 0AEA, 0084,073C,2B64,10C7,00A9,2      ;23191          ALUS_UNSGN OLDALUS?              ;what was previous GPR0?
                        ;23192
                        ;23193
                        ;23194 =10
                        ;23195          ;10-------------------------------;
                        ;23196          VA_R[R3]_RB+CONX(4),             ;GPR0 was nonzero last time thru loop,
                        ;23197          NEXT/CS.M3.WR.L.LOP,             ;loop back
U 0A92, 0884,073D,02E4,C4A7,00A5,8      ;23198          MM.ALLOW.INT?
                        ;23199
                        ;23200          ;11-------------------------------;
U 0A93, 0C84,073D,0024,C4A7,00AE,C      ;23201          VA_R[R3]_RB+CONX(4)              ;GPR0 was zero, we may be done
                        ;23202
                        ;23203          ;--------------------------------;
                        ;23204          M[TEMP0]_MB.AND.ZLIT0[3],        ;Do we have any extra BYTEs to move?
U 0AEC, 0D86,0C12,0A70,1847,00B4,0      ;23205          WX.NE.0?
                        ;23206
                        ;23207 =0
                        ;23208 CS.M3.R1_PC:
                        ;23209          ;0--------------------------------;
                        ;23210          R[R1]_M[PC],                     ;(No, )GPR1_PC, Clear GPR0 & split on
U 0B40, 0885,A592,4034,4047,00B4,8      ;23211          NEXT/CS.MOV.BUT.FLAG1            ;opcode
                        ;23212
                        ;23213          ;1--------------------------------;
                        ;23214          R[R0]_D_M[TEMP0],                ;Yes, GPR0_1, 2, or 3, move BYTEs
U 0B41, 0C84,0592,6034,0047,00A9,7      ;23215          NEXT/CS.M3.WR.LB
```

```
                                ;23216 .TOC     ''      Character String                : MOVC5''
                                ;23217
                                ;23218 =0
                                ;23219 CS.MOVTC.TUC.FIN:
                                ;23220          ;0------------------------------;
                                ;23221          IR<2-0>?,                       ;Which instruction are we? (forward)
U 0B42, 0880,0036,4670,0047,00A6,E  ;23222      NEXT/CS.MOVTC.TUC.TMP10-0
                                ;23223
                                ;23224          ;1------------------------------;
U 0B43, C080,A592,46F0,0047,00AB,1  ;23225      WB_M[TEMP10],WB<31-30>?         ;MTEMP10 < 0?
                                ;23226
                                ;23227 =01
                                ;23228 CS.M5:
                                ;23229          ;01-----------------------------;
                                ;23230          R[R0]_M[TEMP10],                ;GPR0_MTEMP10, finish inst.
U 0AB1, 0884,A592,4034,0047,00AA,D  ;23231      NEXT/CS.MOV.R4.GETS.0
                                ;23232
                                ;23233          ;11-----------------------------;
U 0AB3, 0C84,A593,2034,0047,00AF,4  ;23234      R[R0]_D_-M[TEMP10]              ;R0_D_-MTMP10
                                ;23235
                                ;23236          ;-------------------------------;
U 0AF4, 0586,CC37,0030,4847,00AF,E  ;23237      M[FPDOFFSET]_ZLIT0[9]           ;Pack up instead of RETURN+4
                                ;23238
                                ;23239
                                ;23240          ;*********************************************************************
                                ;23241          ; Here for MOVC5 to fill the extra destination with the FILL character.
                                ;23242          ;*********************************************************************
                                ;23243
                                ;23244          ;-------------------------------;
U 0AFE, 050E,6DB7,0030,1847,00B0,3  ;23245      M[TEMP6]_ZLIT8[3],CLEAR FLAG1   ;Load 300 code, clear MOVC5 flag
                                ;23246
                                ;23247          ;-------------------------------;
U 0B03, 0C80,05BE,43B4,C4A7,00B4,4  ;23248      WB<1-0>.NE.0?,VA_R[R3]          ;Is address LONGWORD aligned?
                                ;23249
                                ;23250 =0
                                ;23251 CS.M5.WR:
                                ;23252          ;0------------------------------;
                                ;23253          WB_R[R0]-CONX(4),WB<31-30>?,    ;Yes, do we have >= 1 LONGWORD left?
U 0B44, 0480,073C,06F4,0047,08A8,5 376*  ;23254 NEXT/CS.M5.WR.L
                                ;23255
                                ;23256 CS.M5.WR.B:
                                ;23257          ;1------------------------------;
U 0B45, 0080,5592,4000,05D8,00B0,4  ;23258      WRITE M[TEMP5],SIZE[BYTE]       ;No, WRITE a BYTE of fill
                                ;23259
                                ;23260          ;-------------------------------;
U 0B04, 0884,0E7D,2A34,0047,08B4,6 378*  ;23261 R[R0]_D_RB-1,WX.EQ.0?            ;GPR0_DREG_length-1, are we finished?
                                ;23262
                                ;23263 =0
                                ;23264          ;0------------------------------;
                                ;23265          VA_R[R3]_RB+1,WB<1-0>.NE.0?,    ;No, inc. dest. address, are we LONGWORD
U 0B46, 0484,0E7C,03B4,C4A7,08B4,4 374*  ;23266 NEXT/CS.M5.WR                    ;aligned now?
                                ;23267
                                ;23268          ;1------------------------------;
                                ;23269          VA_R[R3]_RB+1,                  ;Yes, inc. address, wrap up inst.
U 0B47, 0884,0E7C,0034,C4A7,00AA,D  ;23270      NEXT/CS.MOV.R4.GETS.0
```

```
                                ;23271          ;*****************************************************************************
                                ;23272          ; MOVC5 fill write -- dest. is LONGWORD aligned, =01 is >= 1 LONGWORD
                                ;23273          ; left, =11 is < 1 LONGWORD left.
                                ;23274          ;*****************************************************************************
                                ;23275 =01
                                ;23276 CS.M5.WR.L:
                                ;23277          ;01-------------------------------;
                                ;23278          M[TEMP9]_ZLIT0[3],              ;Set FPD GPR0 decrement flag
U 0AB5, 0586,9C37,0030,1847,00B0,6 ;23279       NEXT/CS.M5.WR.L1
                                ;23280
                                ;23281 CS.M5.WR.LB:
                                ;23282          ;11-------------------------------;
U 0AB7, 0086,95B7,0030,0047,00B4,5 ;23283       M[TEMP9]_0,NEXT/CS.M5.WR.B      ;Clear FPD GPR0 decrement flag, READ/
                                ;23284                                          ;WRITE BYTEs
                                ;23285
                                ;23286 CS.M5.WR.L1:
                                ;23287          ;-------------------------------;
U 0B06, 0086,05BE,4034,0047,00B0,8 ;23288       M[TEMP0]_R[R0]                  ;<1-0> has number of extra BYTEs (if
                                ;23289                                          ;any) to move after moving the LONGWORDs
                                ;23290
                                ;23291          ;-------------------------------;
U 0B08, 0884,9003,C034,0047,00B0,A ;23292       R[R0]_NOT(M[TEMP9]).AND.RB      ;Make GPR0 evenly divisible by 4
                                ;23293
                                ;23294          ;-------------------------------;
                                ;23295          R[R0]_RB-CONX(4),               ;Dec. GPR0 & latch polarity into ALUS,
U 0B0A, 0884,073C,02E4,10C7,00AB,8 ;23296       ALUS_UNSGN,MM.ALLOW.INT?        ;anything pending?
                                ;23297
                                ;23298
                                ;23299          ;*****************************************************************************
                                ;23300          ; =00 is nothing pending, =10 is ints. pending or timer service.
                                ;23301          ; IE.SERV.IP.TS2 will RETURN -1, so =01 is a duplicate of =00.
                                ;23302          ;*****************************************************************************
                                ;23303 =00
                                ;23304 CS.M5.WR.L.LOP:
                                ;23305          ;00-------------------------------;
                                ;23306          WRITE M[TEMP5],SIZE[LONG],      ;WRITE LONGWORD of fill
U 0AB8, 0C80,5592,4020,05D8,00B0,F ;23307       NEXT/CS.M5.WR.L.LOP1
                                ;23308
                                ;23309          ;01-------------------------------;
                                ;23310          WRITE M[TEMP5],SIZE[LONG],      ;WRITE LONGWORD of fill
U 0AB9, 0C80,5592,4020,05D8,00B0,F ;23311       NEXT/CS.M5.WR.L.LOP1
                                ;23312
                                ;23313          ;10-------------------------------;
                                ;23314          PUSH,INTPEND OR TIMER?,         ;Something pending, RETURN -1 or
U 0ABA, 0480,0036,4AF0,0047,04F7,0 ;23315       NEXT/IE.SERV.IP.TS2             ;pack up
                                ;23316 =
```

```
                                ;23317 CS.M5.WR.L.LOP1:
                                ;23318          ;-------------------------------;
                                ;23319          R[R0]_D_RB-CONX(4),              ;Dec. length for NEXT pass, what was
U 0B0F, 0084,073C,2B64,10C7,00A9,E ;23320       ALUS_UNSGN OLDALUS?             ;previous length?
                                ;23321
                                ;23322 =10
                                ;23323          ;10-------------------------------;
                                ;23324          VA_R[R3]_RB+CONX(4),            ;> 0, inc. dest. address, loop back
                                ;23325          NEXT/CS.M5.WR.L.LOP,
U 0A9E, 0084,073D,C2E4,C4A7,00AB,8 ;23326       MM.ALLOW.INT?
                                ;23327
                                ;23328          ;11-------------------------------;
U 0A9F, 0484,073D,0024,C4A7,00B1,0 ;23329       VA_R[R3]_RB+CONX(4)             ;= 0, inc. dest. address
                                ;23330
                                ;23331          ;-------------------------------;
                                ;23332          M[TEMP0]_MB.AND.ZLIT0[3],       ;Are we finished?
U 0B10, 0586,0C12,0A70,1847,00B4,8 ;23333       WX.NE.0?
                                ;23334
                                ;23335 =0
                                ;23336 CS.MOV.BUT.FLAG1:
                                ;23337          ;0--------------------------------;
                                ;23338          FLAG1?,R[R0]_D_0,               ;R0 AND D GET A 0
U 0B48, 0C84,05B7,25F4,0047,00AA,D ;23339       NEXT/CS.MOV.R4.GETS.0           ;MOVC5 OR CMPC5?
                                ;23340
                                ;23341          ;1--------------------------------;
                                ;23342          R[R0]_D_M[TEMP0],               ;GPR0_1, 2, or 3, finishing writing
U 0B49, 0484,0592,6034,0047,00AB,7 ;23343       NEXT/CS.M5.WR.LB                ;BYTES
                                ;23344
                                ;23345
                                ;23346          ;*********************************************************************************
                                ;23347          ; Enter below on IR<2-0>? BUT.  =110 is MOVC3, MOVC5, or MOVTC.
                                ;23348          ; =111 is MOVTUC.
                                ;23349          ;*********************************************************************************
                                ;23350 =110
                                ;23351 CS.M3.MTUC:
                                ;23352          ;110------------------------------;
                                ;23353          FLAG1?,R[R0]_D_0,               ;R0 AND D GET A 0
U 0A56, 0C84,05B7,25F4,0047,00AA,D ;23354       NEXT/CS.MOV.R4.GETS.0           ;MOVC5 OR CMPC5?
                                ;23355
                                ;23356          ;111------------------------------;
                                ;23357          WB_M[TEMP10],WR<31-30>?,        ;Is MTEMP10 < 0?
U 0A57, 0080,A592,46F0,0047,00AC,5 ;23358       NEXT/CS.MTUC.END
                                ;23359 =
```

```
;23360 .TOC     "       Character String                : MOVTC and MOVTUC"
;23361
;23362          ;*****************************************************************************
;23363                  MOVTC
;23364                  INPUT           R0              MIN(SRC LGTH,DEST LGTH)
;23365                                  Q                  ''      ''          ''
;23366                                  MTMP8              ''      ''          ''
;23367                                  D
;23368                                  R1              SOURCE ADDRESS
;23369                                  PC                 ''          ''
;23370                                  TEMP2              ''      ''
;23371                                  TEMP3           DESTINATION LENGTH
;23372                                  TEMP5           ESC(EXPANDED)
;23373                                  TEMP7           TABLE ADDRESS
;23374                                  FLAG1           SET(FOR CMPC5 & MOVC5)
;23375                                  MTEMP10         SRC LGTH - DST LGTH
;23376                                  FPDOFFSET       2LIT0[9]
;23377                                  R3              DESTINATION ADDRESS
;23378                                  VA                 ''          ''
;23379                                  R2              DELTA PC'0(8BITS'24BITS)
;23380                                  TEMP6           4(ROTATED LEFT BY 8)
;23381                                  FPD             SET
;23382
;23383                  OUTPUT          R0              # OF UNTRANSLATED BYTES
;23384                                                  IN SRC STR
;23385                                                  (NE.0 IF SRC>DEST LGTH)
;23386                                  R1              ADD OF ONE BYTE BEYOND
;23387                                                  THE LAST BYTE IN SRC
;23388                                                  STRING THAT WAS TRANS
;23389                                  R2              0
;23390                                  R3              ADD OF THE TRANS TABLE
;23391                                  R4              0
;23392                                  R5              ADD OF ONE BYTE BEYOND
;23393                                                  THE DESTINATION STRING
;23394                                  CC              N_SRCLEN LSS DSTLEN
;23395                                                  Z_SRCLEN EQL DESLEN
;23396                                                  V_0
;23397                                                  C_SRCLEN LSSU DSTLEN
;23398
;23399          RESOURCES       SAME AS MOVC5 WITH THE FOLLOWING
;23400                          TEMP7           TABLE ADDRESS
;23401                          RTEMP12         Holds XB data
;23402                          FLAG0           1
;23403                          FLAG3           _1 IF MOV BACK
;23404
;23405          SUBROUTINES     CS.GEN.RESPC
;23406          ;*****************************************************************************
```

```
;23407 ;*******************************************************************************
;23408 ;       MOVTUC  SAME AS MOVTC EXCEPT:
;23409 ;               OUTPUT                  R0              # OF UNTRANSLATED BYTES
;23410 ;                                                       IN SRC STR INC ESC BYTE
;23411 ;                                                       (=0 IF SRC STR TRANS
;23412 ;                                                       W/O ESC
;23413 ;                                       R1              ADD OF EXHAUST OR ESC
;23414 ;                                                       OR OF ONE BYTE BEYOND
;23415 ;                                                       THE LAST BYTE IN SRC
;23416 ;                                                       STRING THAT WAS TRANS
;23417 ;                                       R2              0
;23418 ;                                       R3              ADD OF THE TRANS TABLE
;23419 ;                                       R4              # OF BYTES IN DEST STR
;23420 ;                                       R5              ADD THAT WOULD HAVE
;23421 ;                                                       TRANS BYTE IF NOT FOR
;23422 ;                                                       EXHAUST OR ESC OR ADD
;23423 ;                                                       OF ONE BYTE BEYOND
;23424 ;                                                       THE DESTINATION STRING
;23425 ;                                       CC              N_SRCLEN LSS DSTLEN
;23426 ;                                                       Z_SRCLEN EQL DESLEN
;23427 ;                                                       V_TERMINATED BY ESC
;23428 ;                                                       C_SRCLEN LSSU DSTLEN
;23429 ;               RESOURCES               SAME AS MOVC5 WITH THE FOLLOWING
;23430 ;                                       TEMP7           TABLE ADDRESS
;23431 ;                                       FLAG0           1
;23432 ;                                       FLAG3           1 IF MOV BACK
;23433 ;*******************************************************************************
```

```
                                ;23434 CS.MOVTC.TUC.BACK:
                                ;23435          ;--------------------------------;
                                ;23436          SET FLAG3,MM.ALLOW.INT?,        ;Write back FLAG, load code 6, anything
U 0B12, 0D5E,6DB7,02F0,3047,00AB,C ;23437       M[TEMP6]_ZLIT8[6]               ;pending?
                                ;23438
                                ;23439
                                ;23440          ;*********************************************************************
                                ;23441          ; Here on MM.ALLOW.INT? BUT.
                                ;23442          ; =00 is nothing pending.
                                ;23443          ; =10 is something pending, RETURN -1 or pack up.
                                ;23444          ; =01 is a duplicate of =00 for IE.SERV.IP.TS2 RETURN.
                                ;23445          ;*********************************************************************
                                ;23446 =00
                                ;23447 CS.MTC.WR:
                                ;23448          ;00--------------------------------;
                                ;23449          PC_ZEXT(XB<7-0>)+R[TEMP7],      ;PC_translated character's add.
U 0ABC, 0481,7015,0641,E487,00A6,6 ;23450       IR<2-0>?,NEXT/CS.MTC.VA
                                ;23451
                                ;23452          ;01--------------------------------;
                                ;23453          PC_ZEXT(XB<7-0>)+R[TEMP7],      ;PC_translated character's address
U 0ABD, 0481,7015,0641,E487,00A6,6 ;23454       IR<2-0>?,NEXT/CS.MTC.VA
                                ;23455
                                ;23456          ;10--------------------------------;
                                ;23457          PUSH,INTPEND OR TIMER?,
U 0ABE, 0480,0036,4AF0,0047,04F7,0 ;23458       NEXT/IE.SERV.IP.TS2
                                ;23459 =
                                ;23460
                                ;23461 =110
                                ;23462 CS.MTC.VA:
                                ;23463          ;110-------------------------------;
                                ;23464          R[TEMP12]_XB PC_PC+1,           ;Write a BYTE from the XB
U 0A66, 0485,7592,4003,2047,00B1,E ;23465       NEXT/CS.MTC.FOB.PRE
                                ;23466
                                ;23467          ;111-------------------------------;
                                ;23468          Q_XB-R[TEMP5] PC_PC+1,          ;MOVTUC[2F], get next BYTE & is it
U 0A67, 0C81,7000,1B41,6047,08AC,2 374* ;23469  SIGND CMP?                       ;equal to fill?
                                ;23470
                                ;23471 =10
                                ;23472          ;10--------------------------------;
                                ;23473          WRITE M[TEMP5]+Q,SIZE[BYTE],    ;Write (XB) BYTE,
                                ;23474          FLAG3?,                         ;GO TO COMMON FLOW WITH MOVTC
U 0AC2, 0480,5009,04C0,05D8,00B4,C ;23475       NEXT/CS.MTC.FOB
                                ;23476
                                ;23477
                                ;23478          ;11--------------------------------;
U 0AC3, 0480,A592,46F0,08E7,00AC,5 ;23479       WB_M[TEMP10],WB<31-30>?,SET V   ;PSL<V>_1, MTEMP10 < 0?
```

```
                                ;23480 =01
                                ;23481 CS.MTUC.END:
                                ;23482          ;01------------------------------;
                                ;23483          D_R[R0],                        ;D GETS R0
U 0AC5, 0480,05BE,6034,0047,00B1,C ;23484       NEXT/CS.R0.GETS.R0+10           ;R0_R0+TMP10 FLOW
                                ;23485
                                ;23486          ;11------------------------------;
U 0AC7, 0080,A003,2034,0047,00B4,A ;23487       D_R[R0]-M[TEMP10]               ;D_R0-TMP10
                                ;23488
                                ;23489 =0
                                ;23490 CS.MOVT.R5.GETS.VA:
                                ;23491          ;0------------------------------;
                                ;23492          PUSH,                           ;RET+1
                                ;23493          R[R5]_M[VA],                    ;R5_VA
U 0B4A, 0885,B592,4035,4047,04AA,5 ;23494       NEXT/CS.GEN.RESPC               ;RESTOR PC
                                ;23495
                                ;23496          ;1------------------------------;
U 0B4B, 0884,05B2,4035,0047,00B1,7 ;23497       R[R4]_D                         ;Load GPR4
                                ;23498
                                ;23499          ;------------------------------;
U 0B17, 0CE4,7592,4034,C047,00B1,B ;23500       R[R3]_M[TEMP7],CLEAR FPD        ;GPR3_TEMP7, FPD_0
                                ;23501
                                ;23502
                                ;23503          ;*********************************************************************
                                ;23504          ; Other Character String instructions exit here.
                                ;23505          ;*********************************************************************
                                ;23506
                                ;23507 CS.MTC.R2_0.IRD1:
                                ;23508          ;------------------------------;
U 0B1B, 0084,05B7,0134,8047,003F,9 ;23509       R[R2]_0,IRD1                    ;GPR2_0, **IRD1**
                                ;23510
                                ;23511
                                ;23512 CS.R0.GETS.R0+10:
                                ;23513          ;------------------------------;
                                ;23514          R[R0]_M[TEMP10]+RB,             ;R0_R0+TMP10
U 0B1C, 0484,A001,0034,0047,00B4,A ;23515       NEXT/CS.MOVT.R5.GETS.VA         ;
                                ;23516
                                ;23517
                                ;23518 CS.MTC.FOB.PRE:
                                ;23519          ;------------------------------;
                                ;23520          WRITE R[TEMP12],SIZE[BYTE],     ;Are we writing backwards?
U 0B1E, 0080,05BE,44C3,05D8,00B4,C ;23521       FLAG3?
                                ;23522
                                ;23523 =0
                                ;23524 CS.MTC.FOB:
                                ;23525          ;0------------------------------;
                                ;23526          PC R[R1]_RB+1,                  ;PC,R1_R1+1
U 0B4C, 0C84,0E7C,0034,4487,00B2,3 ;23527       NEXT/CS.R0-1.FOR                ;DEC R0
                                ;23528
                                ;23529          ;1------------------------------;
                                ;23530          R[R0]_D_RB-1,                   ;DEC R0
                                ;23531          WX.EQ.0?,                       ;FINISHED?
U 0B4D, 0084,0E7D,2A34,0047,08B5,0 378* ;23532  NEXT/CS.MOVT.DEC.PC              ;DEC PC
```

```
                                        ;23533 CS.R0-1.FOR:
                                        ;23534          ;--------------------------------;
                                        ;23535          R[R0]_D_RB-1,                   ;R0,D_R0-1
U 0B23, 0084,0E7D,2A34,0047,08B4,E 378* ;23536          WX.EQ.0?                        ;FINISHED?
                                        ;23537
                                        ;23538 =0
                                        ;23539          ;0--------------------------------;
                                        ;23540          VA_R[R3]_RB+1,MM.ALLOW.INT?,    ;VA_GPR3_GPR3+1, anything pending?
U 0B4E, 0C84,0E7C,02F4,C4A7,00AB,C      ;23541          NEXT/CS.MTC.WR
                                        ;23542
                                        ;23543          ;1--------------------------------;
                                        ;23544          VA_R[R3]_RB+1,                  ;VA,R3_R3+1
U 0B4F, 0084,0E7C,0034,C4A7,00B5,1      ;23545          NEXT/CS.MVT.FINISH              ;
                                        ;23546
                                        ;23547 =0
                                        ;23548 CS.MOVT.DEC.PC:
                                        ;23549          ;0--------------------------------;
                                        ;23550          PC_R[TEMP13]_RB-1,              ;PC,TEMP13_TEMP13-1
U 0B50, 0484,0E7D,0033,4487,00B2,4      ;23551          NEXT/CS.MOVT.DEC.VA             ;GOTO DEC VA FLOW
                                        ;23552
                                        ;23553 CS.MVT.FINISH:
                                        ;23554          ;1--------------------------------;
                                        ;23555          VA_R[R3],                       ;VA_R3
                                        ;23556          SET FLAG0,                      ;SET FLAG0 FOR MOVTC,TUC
                                        ;23557          FLAG3?,                         ;FORWARD OR BACK?
U 0B51, 0C40,05BE,44F4,C4A7,00B4,2      ;23558          NEXT/CS.MOVTC.TUC.FIN           ;FINISHED
                                        ;23559
                                        ;23560 CS.MOVT.DEC.VA:
                                        ;23561          ;--------------------------------;
                                        ;23562          VA_M[TEMP4]_MB-1,MM.ALLOW.INT?, ;VA & TEMP4 _TEMP4-1, anything pending?
U 0B24, 0C86,4080,02FD,84A7,00AB,C      ;23563          NEXT/CS.MTC.WR
                                        ;23564
                                        ;23565
                                        ;23566 =110
                                        ;23567 CS.MOVTC.TUC.TMP10-0:
                                        ;23568          ;110------------------------------;
                                        ;23569          WB_M[TEMP10],WB<31-30>?,        ;Is MTEMP10 < 0?
U 0A6E, 0080,A592,46F0,0047,00AB,1      ;23570          NEXT/CS.M5
                                        ;23571
                                        ;23572          ;111------------------------------;
                                        ;23573          WB_M[TEMP10],WB<31-30>?,        ;Is MTEMP10 < 0?
U 0A6F, 0080,A592,46F0,0047,00AC,5      ;23574          NEXT/CS.MTUC.END                ;(goto MOVTUC)
```

```
;23575 .TOC     ''      Character String                : CMPC3 and CMPC5''
;23576
;23577          ;*************************************************************************
;23578          ;       CMPC3
;23579          ;       INPUT           D               LENGTH
;23580          ;                       TEMP1
;23581          ;                       R1              SOURCE ADDRESS
;23582          ;                       PC                 ''      ''
;23583          ;                       TEMP2              ''      ''
;23584          ;                       R3              DESTINATION ADDRESS
;23585          ;                       VA                 ''          ''
;23586          ;                       R2              DELTA PC'0(8BITS'24BITS)
;23587          ;                       TEMP6           1(ROTATED LEFT BY 8)
;23588          ;                       FPD             SET
;23589          ;                       FPDOFFSET       0
;23590          ;
;23591          ;       RESOURCES       R0              LENGTH
;23592          ;                       R2              DELTA PC'0(8BITS'24BITS)
;23593          ;                       R3              DESTINATION ADDRESS
;23594          ;                       D               LENGTH
;23595          ;                       TEMP6           CODE
;23596          ;                       RTEMP13         Holds data for comparison
;23597          ;                       PC              SOURCE ADDRESS
;23598          ;                       FLAG0           0
;23599          ;                       FLAG1           U
;23600          ;                       FLAG3           0
;23601          ;
;23602          ;       OUTPUT          R0              # BYTES LEFT IN SRC STR
;23603          ;                                       INCL BYTE OF INEQUALITY
;23604          ;                       R1              ADD OF BYTE OF INEQUAL
;23605          ;                                       OR OF ONE BYTE BEYOND
;23606          ;                                       THE SOURCE STRING
;23607          ;                       R2              R0
;23608          ;                       R3              ADD OF BYTE OF INEQUAL
;23609          ;                                       OR OF ONE BYTE BEYOND
;23610          ;                                       THE DESTINATION STRING
;23611          ;                       CC              N_1ST BYTE LSS 2ND BYTE
;23612          ;                                       Z_1ST BYTE EQL 2ND BYTE
;23613          ;                                       V_0
;23614          ;                                       C_1ST BYTE LSSU 2ND BYTE
;23615          ;
;23616          ;*************************************************************************
```

```
;23617  ;*************************************************************************
;23618  ;       CMPC5
;23619  ;       INPUT           R0              MIN(SRC LGTH,DEST LGTH)
;23620  ;                       Q                  ''    ''         ''
;23621  ;                       D
;23622  ;                       R1              SOURCE ADDRESS
;23623  ;                       PC                 ''      ''
;23624  ;                       TEMP2              ''      ''
;23625  ;                       TEMP5           FILL(EXPANDED)
;23626  ;                       TEMP3           DESTINATION LENGTH
;23627  ;                       FLAG1           SET(FOR CMPC5 AND MOVC5)
;23628  ;                       MTEMP10         SRC LGTH - DST LGTH
;23629  ;                       FPDOFFSET       0
;23630  ;                       R3              DESTINATION ADDRESS
;23631  ;                       VA                 ''
;23632  ;                       R2              DELTA PC'0(8BITS'24BITS)
;23633  ;                       TEMP6           1(ROTATED LEFT BY 8)
;23634  ;                       FPD             SET
;23635  ;
;23636  ;       OUTPUT          R0              # BYTES LEFT IN SRC STR
;23637  ;                                       INCL BYTE OF INEQUALITY
;23638  ;                       R1              ADD OF BYTE OF INEQUAL
;23639  ;                                       OR OF ONE BYTE BEYOND
;23640  ;                                       THE SOURCE STRING
;23641  ;                       R2              # BYTES LEFT IN DES STR
;23642  ;                                       INCL BYTE OF INEQUALITY
;23643  ;                       R3              ADD OF BYTE OF INEQUAL
;23644  ;                                       OR OF ONE BYTE BEYOND
;23645  ;                                       THE DESTINATION STRING
;23646  ;                       CC              N_1ST BYTE LSS 2ND BYTE
;23647  ;                                       Z_1ST BYTE EQL 2ND BYTE
;23648  ;                                       V_0
;23649  ;                                       C_1ST BYTE LSSU 2ND BYTE
;23650  ;
;23651  ;       RESOURCES       R0              LENGTH
;23652  ;                       R1              SOURCE ADDRESS
;23653  ;                       R2              DELTA PC'0(8BITS'24BITS)
;23654  ;                       R3              DESTINATION ADDRESS
;23655  ;                       D               LENGTH
;23656  ;                       TEMP6           CODE
;23657  ;                       RTEMP13         Holds data for comparison
;23658  ;                       PC              SOURCE ADDRESS
;23659  ;                       VA              DESTINATION ADDRESS
;23660  ;                       FLAG0           0
;23661  ;                       FLAG1           1
;23662  ;                       FLAG3           0
;23663  ;*************************************************************************
```

```
                                        ;23664 .TOC     ''      Character String                :       CMPC3''
                                        ;23665
                                        ;23666 =0
                                        ;23667 CS.C3.1:
                                        ;23668          ;0--------------------------------;
                                        ;23669          R[R3]_M[VA],WB<1-0>.NE.0?,      ;GPR3_VA, is address LONGWORD aligned?
U 0B52, 0C85,B592,43B4,C047,00A7,0      ;23670          NEXT/CS.C3.LOP
                                        ;23671
                                        ;23672 CS.CMP.R1.GETS.PC:
                                        ;23673          ;1--------------------------------;
U 0B53, 0885,A592,45F4,4047,00AC,8      ;23674          R[R1]_M[PC],FLAG1?              ;GPR1_PC, are we CMPC5?
                                        ;23675
                                        ;23676 =00
                                        ;23677 CS.CMP.CLRFPD:
                                        ;23678          ;00-------------------------------;
                                        ;23679          PUSH,CLEAR FPD,                 ;FPD_0, restore PC & RETURN+1
                                        ;23680          M[TEMP0]_R[R2].RR.24,
U 0AC8, 08E6,02B7,0024,8047,04AA,A      ;23681          NEXT/CS.GEN.RESPC.1
                                        ;23682
                                        ;23683          ;01-------------------------------;
                                        ;23684          M[TEMP2]_R[R1],FLAG3?,          ;TEMP2_PC, are we MATCHC?
U 0AC9, 0486,25BE,44F4,4047,00B5,4      ;23685          NEXT/CS.CMP.R2.GETS.D
                                        ;23686
                                        ;23687          ;10-------------------------------;
U 0ACA, 0986,CC37,0030,4047,00B2,6      ;23688          M[FPDOFFSET]_ZLITO[8]           ;Entering seperate flows, pack up
                                        ;23689                                          ;instead of RETURN+4
                                        ;23690 =
                                        ;23691
                                        ;23692          ;--------------------------------;
                                        ;23693          R[R0]_D_M[TEMP10],SIZE[LONG],   ;GPR0_D_MTEMP10, what is polarity
U 0B26, 0884,A592,6B64,0047,08A8,8 351* ;23694          SIGND CMP?,NEXT/CS.C5           ;of result?
                                        ;23695
                                        ;23696 =0
                                        ;23697 CS.CMP.R2.GETS.D:
                                        ;23698          ;0--------------------------------;
U 0B54, 0484,05B2,4134,8047,003F,9      ;23699          R[R2]_D,IRD1                    ;GPR2_D, **IRD1**
                                        ;23700
                                        ;23701          ;1--------------------------------;
                                        ;23702          WB_R[R0],WX.EQ.0?,CCOP1,        ;GPR0=0?, and set CC,
                                        ;23703          SIZE[LONG],SET FPD,             ;FPD was zero but MATCHC isn't finished
U 0B55, 00E8,05BE,4A24,0847,08B7,C 322* ;23704          NEXT/CS.MAT.Q.GETS.Q-R0         ;GO TO MATCHC FLOW
```

```
                                ;23705 .TOC   "      Character String                :              Main loop"
                                ;23706
                                ;23707          ;*************************************************************************
                                ;23708          ; After initialization enter at CS.C3.LOP: on a WB<1-0>.NE.0? BUT.
                                ;23709          ; =000 is source 2 address is LONGWORD aligned, compare in LONGWORDs.
                                ;23710          ; =001 is source 2 address isn't aligned, compare in BYTEs.
                                ;23711          ; The BYTE compare flow compares until src.#2 address (current) is
                                ;23712          ; LONGWORD aligned, at which point it splits into LONGWORD compare
                                ;23713          ; flows.
                                ;23714          ; The LONGWORD compare flow compare until EOS or a miscompare, at which
                                ;23715          ; point it backs up the PC and compares in BYTEs to isolate the
                                ;23716          ; different BYTE.
                                ;23717          ;
                                ;23718          ; Code also enters at CS.C3.LOP.B.INT: on a MM.ALLOW.INT? BUT.
                                ;23719          ; =001 is nothing pending, =011 is timer or ints. pending, RETURN-1
                                ;23720          ; if just timer, else pack up.
                                ;23721          ;*************************************************************************
                                ;23722 =000
                                ;23723 CS.C3.LOP:
                                ;23724          ;000--------------------------;
                                ;23725          WB_R[R0]-CONX(4),WB<31-30>?,    ;Is remaining length >=4?
U 0A70, 0480,073C,06E4,0047,08A7,9 376*  ;23726          NEXT/CS.C3.LOP.L
                                ;23727
                                ;23728 CS.C3.LOP.B.INT:
                                ;23729          ;001--------------------------;
                                ;23730          R[TEMP13]_ZEXT(XB) PC_PC+1,     ;BYTE compare, READ both sources
U 0A71, 0085,759E,4003,6047,00A7,A       ;23731          NEXT/CS.C3.LOP.B
                                ;23732
                                ;23733          ;010--------------------------;
                                ;23734          R[TEMP13]_ZEXT(XB) PC_PC+1,     ;BYTE compare, READ both sources
U 0A72, 0085,759E,4003,6047,00A7,A       ;23735          NEXT/CS.C3.LOP.B
                                ;23736
                                ;23737          ;011--------------------------;
                                ;23738          PUSH,INTPEND OR TIMER?,
U 0A73, 0480,0036,4AF0,0047,04F7,0       ;23739          NEXT/IE.SERV.IP.TS2
                                ;23740
                                ;23741 =101
                                ;23742          ;101--------------------------;
                                ;23743          M[TEMP2]_FLAGS,                 ;BUS exception, pack up
U 0A75, 0886,2036,4030,0387,00BC,A       ;23744          NEXT/CS.CMP.PACK.BEGIN
                                ;23745
                                ;23746          ;110--------------------------;
                                ;23747          M[TEMP2]_FLAGS,                 ;BUS exception, pack up
U 0A76, 0886,2036,4030,0387,00BC,A       ;23748          NEXT/CS.CMP.PACK.BEGIN
                                ;23749
                                ;23750          ;111--------------------------;
                                ;23751          M[TEMP2]_FLAGS,                 ;Ints. pending, pack up
U 0A77, 0886,2036,4030,0387,00BC,A       ;23752          NEXT/CS.CMP.PACK.BEGIN
```

```
                                        ;23753 =0**
                                        ;23754 CS.C3.LOP.B:
                                        ;23755          ;0**------------------------------;
                                        ;23756          READ,SIZE[BYTE],
U 0A7A, 0C80,0036,4000,0050,00B2,8      ;23757          NEXT/CS.C3.LOP.B.A
                                        ;23758
                                        ;23759          ;1**------------------------------;
                                        ;23760          PC_M[PC]-ZLIT0[1],              ;BUS exception, fix PC, pack up
U 0A7E, 0D81,AC10,0030,0C87,00FC,8      ;23761          NEXT/CS.CMP.PACK
                                        ;23762
                                        ;23763
                                        ;23764 CS.C3.LOP.B.A:
                                        ;23765          ;------------------------------;
                                        ;23766          WB_R[TEMP13]-ZEXT(M[MDR]),      ;Set CC on these BYTEs, are they
                                        ;23767          SIZE[BYTE],CCOP1,WX.EQ.0?,      ;equal?
U 0B2B, 0081,2017,0A03,4847,08B5,A 356* ;23768          NEXT/CS.C3.LOP.B1
```

```
                                ;23769 .TOC     "       Character String                :               LONGWORDs"
                                ;23770
                                ;23771          ;*************************************************************************
                                ;23772          ; Beginning of LONGWORD compare.
                                ;23773          ; =001 is (remaining) length >= 4, we can compare LONGWORDs.
                                ;23774          ; =011 is (remaining) length < 4, compare in BYTEs.
                                ;23775          ; We pass thru here only once.  Subsequent LONGWORDs are read in a
                                ;23776          ; different loop at CS.C3.LOP.L1 which includes interrupt pending/
                                ;23777          ; timer service splits.
                                ;23778          ;*************************************************************************
                                ;23779 =001
                                ;23780 CS.C3.LOP.L:
                                ;23781          ;001-------------------------------;
                                ;23782          R[TEMP13]_XB PC_PC+4,           ;READ both sources
U 0A79, 0485,7592,4023,6047,00A8,0 ;23783       NEXT/CS.C3.LOP.L1
                                ;23784
                                ;23785          ;011-------------------------------;
                                ;23786          R[TEMP13]_ZEXT(XB) PC_PC+1,     ;READ both sources, BYTE
U 0A7B, 0085,759E,4003,6047,00A7,A ;23787       NEXT/CS.C3.LOP.B
                                ;23788
                                ;23789 =101
                                ;23790          ;101-------------------------------;
                                ;23791          M[TEMP2]_FLAGS,                 ;BUS exception, pack up
U 0A7D, 0886,2036,4030,0387,00BC,A ;23792       NEXT/CS.CMP.PACK.BEGIN
                                ;23793
                                ;23794 =111
                                ;23795          ;111-------------------------------;
                                ;23796          M[TEMP2]_FLAGS,                 ;BUS exception, pack up
U 0A7F, 0886,2036,4030,0387,00BC,A ;23797       NEXT/CS.CMP.PACK.BEGIN
                                ;23798
                                ;23799
                                ;23800 =0**
                                ;23801 CS.C3.LOP.L1:
                                ;23802          ;0**-----------------------------;
                                ;23803          READ,SIZE[LONG],
U 0A80, 0080,0036,4020,0050,00B2,C ;23804       NEXT/CS.C3.LOP.L1.A
                                ;23805
                                ;23806          ;1**-----------------------------;
                                ;23807          PC_M[PC]-ZLIT0[4],              ;BUS exception, fix PC, pack up
U 0A84, 0181,AC10,0030,2487,00FC,8 ;23808       NEXT/CS.CMP.PACK
```

```
                                ;23809 CS.C3.LOP.L1.A:
                                ;23810          ;-------------------------------;
                                ;23811          WB_R[TEMP13]-M[MDR],CCOP1,      ;Set CC & are these pieces equal?
U 0B2C, 0481,2003,0A23,4847,08B5,6 345* ;23812  SIZE[LONG],WX.EQ.0?
                                ;23813
                                ;23814 =0
                                ;23815          ;0------------------------------;
                                ;23816          PC_M[PC]-ZLIT0[4],              ;No, PC_PC-4 to isolate the BYTE,
                                ;23817          MM.ALLOW.INT?,                  ;anything pending?
U 0B56, 0D81,AC10,02F0,2487,00A7,1      ;23818  NEXT/CS.C3.LOP.B.INT
                                ;23819
                                ;23820          ;1------------------------------;
U 0B57, 0884,073C,2A64,0047,08B5,8 377* ;23821  R[R0]_D_RB-CONX(4),WX.NE.0?       ;Yes, dec. length, are we finished?
                                ;23822
                                ;23823 =0
                                ;23824          ;0------------------------------;
                                ;23825          VA_R[R3]_RB+CONX(4),            ;Yes, bump VA & GPR3
U 0B58, 0C84,073D,0024,C4A7,00B5,3      ;23826  NEXT/CS.CMP.R1.GETS.PC
                                ;23827
                                ;23828          ;1------------------------------;
U 0B59, 0480,073C,06E4,0047,08AC,D 376* ;23829  WB_R[R0]-CONX(4),WB<31-30>?       ;No, Is there >= 1 LONGWORD left?
                                ;23830
                                ;23831 =01
                                ;23832          ;01-----------------------------;
                                ;23833          VA_R[R3]_RB+CONX(4),            ;Yes, compare next LONGWORD, but first
                                ;23834          MM.ALLOW.INT?,                  ;is anything pending?
U 0ACD, 0084,073D,02E4,C4A7,00A8,1      ;23835  NEXT/CS.C3.LOP.L.INT
                                ;23836
                                ;23837          ;11-----------------------------;
                                ;23838          VA_R[R3]_RB+CONX(4),            ;Yes, 1-3 BYTEs are left, bump VA
U 0ACF, 0484,073D,0024,C4A7,00A7,1      ;23839  NEXT/CS.C3.LOP.B.INT
```

```
                                ;23840 =000
                                ;23841 =001
                                ;23842 CS.C3.LOP.L.INT:
                                ;23843          ;001----------------------------;
                                ;23844          R[TEMP13]_XB PC_PC+4,           ;READ both sources
U 0A81, 0485,7592,4023,6047,00A8,0  ;23845      NEXT/CS.C3.LOP.[1
                                ;23846
                                ;23847          ;010----------------------------;
                                ;23848          R[TEMP13]_XB PC_PC+4,           ;READ both sources
U 0A82, 0485,7592,4023,6047,00A8,0  ;23849      NEXT/CS.C3.LOP.[1
                                ;23850
                                ;23851          ;011----------------------------;
                                ;23852          PUSH,INTPEND OR TIMER?,
U 0A83, 0480,0036,4AF0,0047,04F7,0  ;23853      NEXT/IE.SERV.IP.TS2
                                ;23854
                                ;23855 =101
                                ;23856          ;101---------------------------- --;
                                ;23857          M[TEMP2]_FLAGS,                 ;BUS exception, pack up
U 0A85, 0886,2036,4030,0387,00BC,A  ;23858      NEXT/CS.CMP.PACK.BEGIN
                                ;23859
                                ;23860          ;110----------------------------;
                                ;23861          M[TEMP2]_FLAGS,                 ;BUS exception, pack up
U 0A86, 0886,2036,4030,0387,00BC,A  ;23862      NEXT/CS.CMP.PACK.BEGIN
                                ;23863
                                ;23864          ;111----------------------------;
                                ;23865          M[TEMP2]_FLAGS,                 ;Ints. pending, pack up
U 0A87, 0886,2036,4030,0387,00BC,A  ;23866      NEXT/CS.CMP.PACK.BEGIN
```

```
                                ;23867 .TOC     "       Character String                :               BYTEs"
                                ;23868
                                ;23869          ;*************************************************************************
                                ;23870          ; Here from BYTE compare loop.  =0 is BYTEs are not equal, =1 is equal.
                                ;23871          ;*************************************************************************
                                ;23872 =0
                                ;23873 CS.C3.LOP.B1:
                                ;23874          ;0-------------------------------;
                                ;23875          PC_R[R1]_M[PC]-1,               ;PC,R1,_PC-1
                                ;23876          FLAG1?,                         ;CMPC5?
0 0B5A, 0885,AE51,05F4,4487,00A6,0              ;23877          NEXT/CS.C3.CLRFPD.1
                                ;23878
                                ;23879          ;1-------------------------------;
0 0B5B, 0084,0E7D,2A34,0047,08B5,C 378*         ;23880          R[R0]_D_RB-1,WX.EQ.0?           ;Dec. length, are we finished?
                                ;23881
                                ;23882 =0
                                ;23883          ;0-------------------------------;
                                ;23884          R[R3]_VA_RB+1,WB<1-0>.NE.0?,    ;No, GPR3_VA_address+1, are we aligned?
0 0B5C, 0484,0E7C,03B4,C4A7,08A7,0 374*         ;23885          NEXT/CS.C3.LOP
                                ;23886
                                ;23887          ;1-------------------------------;
                                ;23888          VA_R[R3]_RB+1,                  ;Yes, VA,R3_R3+1
0 0B5D, 0884,0E7C,0034,C4A7,00B5,3              ;23889          NEXT/CS.CMP.R1.GETS.PC          ;GO to R1_PC FLOW
                                ;23890
                                ;23891 =0*
                                ;23892 CS.C3.CLRFPD.1:
                                ;23893          ;0*------------------------------;
0 0A60, 0C80,0036,4030,0047,00AC,8              ;23894          NEXT/CS.CMP.CLRFPD              ;GOTO FPD FLOW
                                ;23895
                                ;23896          ;1*------------------------------;
                                ;23897          R[TEMP4]_M[TEMP10],             ;TEMP4_MTEMP10, is it <0?
0 0A62, 0084,A592,46F1,0047,00AD,1              ;23898          WB<31-30>?
                                ;23899
                                ;23900 =01
                                ;23901          ;01------------------------------;
                                ;23902          CLEAR FLAG1,                    ;CLR CMPC5
                                ;23903          R[R0]_M[TEMP10]+RB,             ;R0_R0-MTMP10
0 0AD1, 0C0C,A001,0034,0047,00B5,3              ;23904          NEXT/CS.CMP.R1.GETS.PC          ;MTMP10 > = 0
                                ;23905
                                ;23906          ;11------------------------------;
                                ;23907          D_D-R[TEMP4],                   ;D_D-MTMP10(TMP4)
                                ;23908          CLEAR FLAG1,                    ;CLR CMPC5
0 0AD3, 0C08,0020,2031,0047,00AC,8              ;23909          NEXT/CS.CMP.CLRFPD              ;MTMP10 < 0
```

```
                                ;23910 .TOC     "       Character String                :       CMPC5"
                                ;23911
                                ;23912          ;*************************************************************************
                                ;23913          ; Here from SIGND CMP? split on (Src.#1 length) - (Src.#2 length).
                                ;23914          ; =000 is Src.#1 is longer, compare the excess with the FILL character.
                                ;23915          ; =001 is Sources are equal in length, finish inst.
                                ;23916          ; =010 is Src.#2 is longer, compare the excess with the FILL char.
                                ;23917          ;*************************************************************************
                                ;23918 =000
                                ;23919 CS.C5:
                                ;23920          ;000-------------------------------;
                                ;23921          PUSH,NEXT/CS.C5.S1F,            ;RETURN +3
U 0A88, 0186,6DB7,0030,1047,04B3,7 ;23922       M[TEMP6]_ZLIT8[2]
                                ;23923
                                ;23924          ;001-------------------------------;
U 0A89, 0C80,0036,4030,0047,00AC,8 ;23925       NEXT/CS.CMP.CLRFPD              ;TMP10=0-NO MORE CMP-CLEAN UP
                                ;23926
                                ;23927          ;010-------------------------------;
U 0A8A, 0081,8002,403D,8487,00B2,E ;23928       PC_M[VA],NEXT/CS.C5.S2F         ;PC_VA
                                ;23929
                                ;23930          ;011-------------------------------;
                                ;23931          CLEAR FPD,PUSH,                 ;Finished with extra source 1 compare,
                                ;23932          M[TEMP0]_R[R2].RR.24,           ;restore PC
U 0A8B, 08E6,02B7,0024,8047,04AA,A ;23933       NEXT/CS.GEN.RESPC.1
                                ;23934
                                ;23935          ;100-------------------------------;
                                ;23936          M[TEMP2]_R[R1],FLAG3?,          ;TEMP2_PC, are we MATCHC?
U 0A8C, 0486,25BE,44F4,4047,00B5,4 ;23937       NEXT/CS.CMP.R2.GETS.D
                                ;23938 =
                                ;23939
                                ;23940
                                ;23941 CS.C5.S2F:
                                ;23942          ;--------------------------------;
U 0B2F, 0884,05B3,2034,0047,00B3,3 ;23943       R[R0]_D_-D                      ;Get -(-(extra length to compare))
                                ;23944
                                ;23945          ;--------------------------------;
                                ;23946          M[TEMP6]_ZLIT8[4],              ;TEMP6_400, enter src.#2 compare
U 0B33, 0D86,6DB7,0030,2047,00B9,9 ;23947       NEXT/CS.C5.S2F.1
```

```
                                ;23946 .TOC     "       Character String                :               Source 1 - FILL"
                                ;23949
                                ;23950          ;*************************************************************************
                                ;23951          ; Here for Source 1 - FILL compare.  This routine is used by SKPC.
                                ;23952          ; Enter at CS.C5.S1F and RETURN +3.
                                ;23953          ;*************************************************************************
                                ;23954
                                ;23955 CS.C5.S1F:
                                ;23956          ;-------------------------------;
U 0B37, 0180,0C30,06F0,2047,08AD,5 377*  ;23957          WB_D-ZLIT0[4],WB<31-30>?        ;Is there >= 1 LONGWORD left in string?
                                ;23958
                                ;23959          ;*************************************************************************
                                ;23960          ; =01 is >= 1 LONGWORD in string, compare LONGWORDs.
                                ;23961          ; =11 is < 1 LONGWORD left, compare BYTEs.
                                ;23962          ; =10 is after comparing a BYTE, finding it equal to the FILL, and
                                ;23963          ; reaching EOS.
                                ;23964          ;*************************************************************************
                                ;23965 =00
                                ;23966 =01
                                ;23967 CS.C5.S1F.LOP:
                                ;23968          ;01-----------------------------;
                                ;23969          WB_M[PC],WB<1-0>.NE.0?,         ; LONGWORD ALIGNED?
U 0AD5, 0881,A592,43B0,0047,0147,C       ;23970          NEXT/CS.C5.S1F.LOP.-1           ;
                                ;23971
                                ;23972 CS.C5.S1F.B1:
                                ;23973          ;10-----------------------------;
U 0AD6, 0485,A592,40B4,4047,0000,3       ;23974          R[R1]_M[PC],RETURN [3]          ;Finished READing BYTEs, RETURN
                                ;23975
                                ;23976 CS.C5.S1F.B:
                                ;23977          ;11-----------------------------;
                                ;23978          WB_ZEXT(XB)-R[TEMP5] PC_PC+1,   ;Compare first/next BYTE with FILL
                                ;23979          CCOP1 SIGND?,                   ;
U 0AD7, 0C81,7014,0B41,6847,08AD,A 385*  ;23980          NEXT/CS.C5.S1F.BC               ;
                                ;23981 147C:
                                ;23982 CS.C5.S1F.LOP.-1:
                                ;23983          ;0******FORCE ADDRESS**********;
                                ;23984          WB_XB-R[TEMP5] PC_PC+4,CCOP1,   ;Compare with FILL char.
U 147C, 0081,7000,0A21,6847,08B5,E 345*  ;23985          WX.EQ.0?,NEXT/CS.C5.S1F.L
                                ;23986 147D:
                                ;23987          ;1******FORCE ADDRESS**********;
                                ;23988          WB_ZEXT(XB)-R[TEMP5] PC_PC+1,   ; COMPARE FIRST/NEXT BYET WITH FILL
U 147D, 0481,7014,0B41,6847,0973,A 385*  ;23989          CCOP1 SIGND?                    ;
                                ;23990 173A:
                                ;23991          ;10*****FORCE ADDRESS**********;
                                ;23992          D_R[ZERO],                      ; BYTE IS DIFFERENT, LOAD PC-1
U 173A, 0080,05BE,603D,8047,00B9,8       ;23993          NEXT/CS.C5.S1F.R1_PC-1  ; AND RETURN+3
                                ;23994
                                ;23995 173B:
                                ;23996          ;11*****FORCE ADDRESS**********;
U 173B, 0484,0E71,20B4,0047,0000,0       ;23997          R[R0]_D_D-1,RETURN [0]          ; TRY AGAIN
                                ;23998 =10
                                ;23999 CS.C5.S1F.BC:
                                ;24000          ;10-----------------------------;
                                ;24001          D_R[ZERO],                      ;BYTEs are different, load PC-1
U 0ADA, 0080,05BE,603D,8047,00B9,8       ;24002          NEXT/CS.C5.S1F.R1_PC-1          ;and RETURN+3
```

```
                                ;24003          ;11------------------------------;
                                ;24004          R[R0]_D_D-1,WX.NE.0?,            ;BYTEs are equal, are we done?
U 0ADB, 0484,0E71,2A74,0047,08AD,6 372*  ;24005          NEXT/CS.C5.S1F.B1
                                ;24006
                                ;24007
                                ;24008 CS.C5.S1F.R1_PC-1:
                                ;24009          ;--------------------------------;
U 0B98, 0485,AE51,00B4,4047,0000,3       ;24010          R[R1]_M[PC]-1,RETURN [3]
                                ;24011 =0
                                ;24012 CS.C5.S1F.L:
                                ;24013          ;0-------------------------------;
                                ;24014          PC_M[PC]-ZLIT0[4],               ;LONGWORD is unequal to FILL,
U 0B5E, 0981,AC10,0030,2487,00AD,7       ;24015          NEXT/CS.C5.S1F.B                 ;back up PC and find the first unequal
                                ;24016                                           ;BYTE
                                ;24017
                                ;24018          ;1-------------------------------;
U 0B5F, 0084,0730,2A64,0047,08B6,0 377*  ;24019          R[R0]_D_D-CONX(4),WX.NE.0?        ;LONGWORD is equal to FILL, dec. length,
                                ;24020                                           ;are we finished?
                                ;24021
                                ;24022 =0
                                ;24023          ;0-------------------------------;
U 0B60, 0485,A592,40B4,4047,0000,3       ;24024          R[R1]_M[PC],RETURN [3]           ;Yes, exit routine
                                ;24025
                                ;24026          ;1-------------------------------;
U 0B61, 0C80,073C,06E4,0047,08AD,D 376*  ;24027          WB_R[R0]-CONX(4),WB<31-30>?      ;No, is there >= 1 LONGWORD left?
                                ;24028
                                ;24029
                                ;24030          ;*****************************************************************************
                                ;24031          ; =01 is GPR0 > 3, loop back for more LONGWORDs.
                                ;24032          ; =11 is GPR0 < 4, 1-3 BYTEs are remaining to be compared.
                                ;24033          ;*****************************************************************************
                                ;24034 =01
                                ;24035          ;01------------------------------;
U 0ADD, 0080,0036,4B30,18E7,00B6,2       ;24036          NEXT/CS.C5.S1F.L.INT,IP.TS?      ;Anything pending before we loop thru?
                                ;24037
                                ;24038          ;11------------------------------;
                                ;24039          WB_ZEXT(XB)-R[TEMP5] PC_PC+1,    ;Compare next BYTE with FILL
                                ;24040          CCOP1 SIGND?,
U 0ADF, 0C81,7014,0B41,6847,08AD,A 385*  ;24041          NEXT/CS.C5.S1F.BC
                                ;24042
                                ;24043
                                ;24044 =0
                                ;24045 CS.C5.S1F.L.INT:
                                ;24046          ;0-------------------------------;
                                ;24047          WB_XB-R[TEMP5] PC_PC+4,CCOP1,    ;No, or timer service, compare next
U 0B62, 0081,7000,0A21,6847,08B5,E 345*  ;24048          WX.EQ.0?,NEXT/CS.C5.S1F.L        ;LONGWORD with the FILL char.
                                ;24049
                                ;24050          ;1-------------------------------;
                                ;24051          PUSH,INTPEND OR TIMER?,          ;Yes, RETURN-1 if timer, pack up if
U 0B63, 0480,0036,4AF0,0047,04F7,0       ;24052          NEXT/IE.SERV.IP.TS2              ;interrupt
```

```
                                ;24053 .TOC     "       Character String                :               Source 2 - FILL"
                                ;24054
                                ;24055          ;*************************************************************************
                                ;24056          ; Here for Source 2 - FILL compare.
                                ;24057          ;*************************************************************************
                                ;24058
                                ;24059 CS.C5.S2F.1:
                                ;24060          ;-----------------------------;
                                ;24061          WB_D-ZLITO[4],WB<31-30>?,       ;Is there >= 1 LONGWORD left in string?
U 0B99, 0908,0C30,06F0,2047,08AE,1 377*  ;24062          CLEAR FLAG1
                                ;24063
                                ;24064          ;*************************************************************************
                                ;24065          ; =01 is >= 1 LONGWORD in string, compare LONGWORDs.
                                ;24066          ; =11 is < 1 LONGWORD left, compare BYTEs.
                                ;24067          ; =10 is after comparing a BYTE, finding it equal to the FILL, and
                                ;24068          ; reaching EOS.
                                ;24069          ;*************************************************************************
                                ;24070 =00
                                ;24071 =01
                                ;24072          ;01---------------------------;
                                ;24073          WB_R[TEMP5]-XB PC_PC+4,CCOP1,   ;Compare with FILL char.
U 0AE1, 0081,7003,0A21,6847,08B6,4 345*  ;24074          WX.EQ.0?,NEXT/CS.C5.S2F.L
                                ;24075
                                ;24076 CS.C5.S2F.B1:
                                ;24077          ;10---------------------------;
U 0AE2, 0C85,A592,4034,C047,00B9,A       ;24078          R[R3]_M[PC],NEXT/CS.C5.S2F.R0_0 ;Finished READing BYTEs, exit loop
                                ;24079
                                ;24080 CS.C5.S2F.B:
                                ;24081          ;11---------------------------;
                                ;24082          WB_R[TEMP5]-ZEXT(XB) PC_PC+1,   ;Compare first/next BYTE with FILL
U 0AE3, 0C81,7017,0B41,6847,08AE,6 385*  ;24083          CCOP1 SIGND?
                                ;24084
                                ;24085 =10
                                ;24086 CS.C5.S2F.BC:
                                ;24087          ;10---------------------------;
                                ;24088          R[R3]_M[PC]-1,                  ;BYTEs are different, load PC-1 and
U 0AE6, 0C85,AE51,0034,C047,00B9,A       ;24089          NEXT/CS.C5.S2F.R0_0             ;exit loop
                                ;24090
                                ;24091          ;11---------------------------;
                                ;24092          R[R0]_D_D-1,WX.NE.0?,            ;BYTEs are equal, are we done?
U 0AE7, 0C84,0E71,2A74,0047,08AE,2 372*  ;24093          NEXT/CS.C5.S2F.B1
                                ;24094
                                ;24095
                                ;24096 CS.C5.S2F.R0_0:
                                ;24097          ;---------------------------;
U 0B9A, 0884,05B7,0034,0047,00AC,8       ;24098          R[R0]_0,NEXT/CS.CMP.CLRFPD      ;GPR0_0, exit loop & finish inst.
```

```
                                ;24099 =0
                                ;24100 CS.C5.S2F.L:
                                ;24101          ;0--------------------------------;
                                ;24102          PC_M[PC]-ZLIT0[4],              ;LONGWORD is unequal to FILL,
U 0B64, 0181,AC10,0030,2487,00AE,3  ;24103      NEXT/CS.C5.S2F.B                ;back up PC and find the first unequal
                                ;24104                                          ;BYTE
                                ;24105
                                ;24106          ;1--------------------------------;
U 0B65, 0084,0730,2A64,0047,08B6,6 377*  ;24107 R[R0]_D_D-CONX(4),WX.NE.0?        ;LONGWORD equal to FILL, are we
                                ;24108                                          ;finished?
                                ;24109
                                ;24110 =0
                                ;24111          ;0--------------------------------;
U 0B66, 0C85,A592,4034,C047,00B9,A  ;24112      R[R3]_M[PC],NEXT/CS.C5.S2F.R0_0  ;Yes
                                ;24113
                                ;24114          ;1--------------------------------;
U 0B67, 0480,073C,06E4,0047,08AE,9 376*  ;24115 WB_R[R0]-CONX(4),WB<31-30>?       ;No, is there >= 1 LONGWORD left?
                                ;24116
                                ;24117
                                ;24118          ;*****************************************************************************
                                ;24119          ; =01 is GPR0 > 3, continue comparing LONGWORDs.
                                ;24120          ; =11 is GPR0 < 4, we have 1, 2, or 3 BYTEs left to compare.  Enter
                                ;24121          ;   BYTE compare loop.
                                ;24122          ;*****************************************************************************
                                ;24123 =01
                                ;24124          ;01-------------------------------;
U 0AE9, 0080,0036,4B30,18E7,00B6,8  ;24125      NEXT/CS.C5.S2F.L.INT,IP.TS?      ;Anything pending before we loop back?
                                ;24126
                                ;24127          ;11-------------------------------;
                                ;24128          WB_R[TEMP5]-ZEXT(XB) PC_PC+1,    ;Compare next BYTE with FILL
                                ;24129          CCOP1 SIGND?,
U 0AEB, 0C81,7017,0B41,6847,08AE,6 385*  ;24130 NEXT/CS.C5.S2F.BC
                                ;24131
                                ;24132 =0
                                ;24133 CS.C5.S2F.L.INT:
                                ;24134          ;0--------------------------------;
                                ;24135          WB_R[TEMP5]-XB PC_PC+4,CCOP1,    ;No, or timer service,  ompare next
U 0B68, 0081,7003,0A21,6847,08B6,4 345*  ;24136 WX.EQ.0?,NEXT/CS.C5.S2F.L         ;LONGWORD with the FILI  char.
                                ;24137
                                ;24138          ;1--------------------------------;
                                ;24139          PUSH,INTPEND OR TIMER?,          ;Yes, RETURN-1 if time:  pack up if
U 0B69, 0480,0036,4AF0,0047,04F7,0  ;24140      NEXT/IE.SERV.IP.TS2              ;interrupt
```

```
;24141 .TOC     ''       Character String                : SCANC and SPANC''
;24142
;24143          ;*************************************************************************
;24144          ;       SPANC and SCANC
;24145          ;
;24146          ;       INPUT           R0              LENGTH
;24147          ;                       D                 ''
;24148          ;                       Q                 ''
;24149          ;                       R1              ADDRESS
;24150          ;                       PC                ''
;24151          ;                       R3              TABLE ADDRESS
;24152          ;                       TEMP5           MASK
;24153          ;                       R2              ORIGINAL PC
;24154          ;                       FPD             SET
;24155          ;
;24156          ;       OUTPUT          FPDOFFSET       5
;24157          ;                       R0              # BYTES LEFT IN STR
;24158          ;                                       INCL NONZERO .AND. BYTE
;24159          ;                       R1              ADD OF BYTE OF NONZERO
;24160          ;                                       .AND. OR ONE BYTE BEYOND
;24161          ;                                       THE STRING
;24162          ;                       R2              R0
;24163          ;                       R3              ADDRESS OF TABLE
;24164          ;                       CC              NZVC_0100 IF R0=0
;24165          ;                                       NZVC_0000 ELSE
;24166          ;
;24167          ;       RESOURCES       R0              LENGTH
;24168          ;                       R1              ADDRESS
;24169          ;                       R2              ORIGINAL PC
;24170          ;                       R3              TABLE ADDRESS
;24171          ;                       D               LENGTH
;24172          ;                       TEMP5           CHARACTER
;24173          ;                       FPDOFFSET       (FOR CMP'S = 5)
;24174          ;                       VA
;24175          ;                       PC              ADDRESS
;24176          ;                       FLAG0           0
;24177          ;                       FLAG1           0
;24178          ;                       FLAG3           0
;24179          ;*************************************************************************
```

```
                                ;24180          ;*************************************************************************
                                ;24181          ; Enter at CS.SCSP from instruction initialization.  Each loop
                                ;24182          ; iteration splits to either CS.SCSP.RO.INC: (finished) or
                                ;24183          ; CS.SCSP.  From CS.SCSP we split on MM.ALLOW.INT?, =00 is nothing
                                ;24184          ; pending, =10 is something pending, RETURN -1 if timer else pack up.
                                ;24185          ;*************************************************************************
                                ;24186 =0
                                ;24187 CS.SCSP.RO.INC:
                                ;24188          ;0-----------------------------;
                                ;24189          R[RO]_D_RB+1,CCOP2,             ;GPRO_D_GPRO+1, set CC
                                ;24190          SIZE[LONG],
U 0B6A, 0484,0E7C,2024,1047,00B9,B  ;24191      NEXT/CS.PC.GETS.PC-1.SCAN
                                ;24192
                                ;24193 CS.SCSP:
                                ;24194          ;1-----------------------------;
                                ;24195          VA_ZEXT(XB)+R[R3] PC_PC+1,      ;Load table address + offset
U 0B6B, 0451,7015,02C4,E4A7,00AE,D  ;24196      SET FLAG2,MM.ALLOW.INT?         ;SET FLAG FOR POSSIBLE PACKING
                                ;24197
                                ;24198 =00
                                ;24199 =01
                                ;24200          ;01----------------------------;
                                ;24201          READ,SIZE[BYTE],CLEAR FLAG2,    ;READ table BYTE
U 0AED, 0810,0036,4640,0050,00AF,2  ;24202      IR<2-0>?,NEXT/CS.SCSP.SPL
                                ;24203
                                ;24204          ;10----------------------------;
                                ;24205          READ,SIZE[BYTE],CLEAR FLAG2,    ;READ table BYTE
U 0AEE, 0810,0036,4640,0050,00AF,2  ;24206      IR<2-0>?,NEXT/CS.SCSP.SPL
                                ;24207
                                ;24208          ;11----------------------------;
                                ;24209          PUSH,INTPEND OR TIMER?,
U 0AEF, 0480,0036,4AF0,0047,04F7,0  ;24210      NEXT/IE.SERV.IP.TS2
                                ;24211
                                ;24212 =*10
                                ;24213 CS.SCSP.SPL:
                                ;24214          ;*10---------------------------;
                                ;24215          R[RO]_D_RB-1,CCOP2,SIZE[WORD],  ;Set CC, dec length, are we finished?
U 0AF2, 0084,0E7D,2A14,1047,08B6,C 378*  ;24216  WX.EQ.0?,NEXT/CS.MDR.MASK.SCAN
                                ;24217
                                ;24218          ;*11---------------------------;
                                ;24219          R[RO]_D_RB-1,CCOP2,SIZE[WORD],  ;GPRO_D_GPRO-1, set CC,
                                ;24220          WX.EQ.0?,                       ;is this last BYTE?
U 0AF3, 0884,0E7D,2A14,1047,08B7,0 378*  ;24221  NEXT/CS.MDR.MASK.SPAN
```

```
                                        ;24222 =0
                                        ;24223 CS.MDR.MASK.SCAN:
                                        ;24224          ;0--------------------------------;
                                        ;24225          WB_ZEXT(M[MDR]).AND.R[TEMP5],   ;AND BYTE WITH THE MASK
                                        ;24226          SIZE[BYTE],                     ;ZEXT BY A BYTE
                                        ;24227          WX.EQ.0?,                       ;IS THE RESULT OF THE AND =0?
U 0B6C, 0C81,2016,0A01,4047,00B6,A      ;24228          NEXT/CS.SCSP.R0.INC
                                        ;24229
                                        ;24230          ;1--------------------------------;
                                        ;24231          WB_ZEXT(M[MDR]).AND.R[TEMP5],   ;AND BYTE WITH MASK
                                        ;24232          SIZE[BYTE],                     ;ZEXT BY A BYTE
U 0B6D, 0481,2016,0A01,4047,00B6,E      ;24233          WX.EQ.0?                        ;=0?
                                        ;24234
                                        ;24235 =0
                                        ;24236 CS.R0.D.GETS.RB+1.SC.SP:
                                        ;24237          ;0--------------------------------;
                                        ;24238          R[R0]_D_RB+1,                   ;INCREMENT R0
                                        ;24239          CCOP2,                          ;SET CC
                                        ;24240          SIZE[LONG],                     ;SO CCOP2 DOESN'T GET A VALIDITY CHECK
U 0B6E, 0484,0E7C,2024,1047,00B9,B      ;24241          NEXT/CS.PC.GETS.PC-1.SCAN       ;PC_PC-1-GOTO COMMON FLOW
                                        ;24242
                                        ;24243 CS.R1.GETS.PC.SCAN:
                                        ;24244          ;1--------------------------------;
                                        ;24245          R[R1]_M[PC],                    ;GPR1_PC, finish instruction
U 0B6F, 0885,A592,4034,4047,00AD,9      ;24246          NEXT/CS.SCSP.EXIT
                                        ;24247
                                        ;24248 CS.PC.GETS.PC-1.SCAN:
                                        ;24249          ;--------------------------------;
                                        ;24250          PC_M[PC]-ZLIT0[1],              ;PC_PC-1
U 0B9B, 0D81,AC10,0030,0C87,00B6,F      ;24251          NEXT/CS.R1.GETS.PC.SCAN         ;
                                        ;24252
                                        ;24253 =0
                                        ;24254 CS.MDR.MASK.SPAN:
                                        ;24255          ;0--------------------------------;
                                        ;24256          WB_ZEXT(M[MDR]).AND.R[TEMP5],   ;MDR.AND.MASK
                                        ;24257          SIZE[BYTE],                     ;ZEXT BY A BYTE
                                        ;24258          WX.NE.0?,                       ;NOT = 0?
U 0B70, 0881,2016,0A41,4047,00B6,A      ;24259          NEXT/CS.SCSP.R0.INC
                                        ;24260
                                        ;24261          ;1--------------------------------;
                                        ;24262          WB_ZEXT(M[MDR]).AND.R[TEMP5],   ;MDR.AND.MASK
                                        ;24263          SIZE[BYTE],                     ;ZEXT BY A BYTE
                                        ;24264          WX.NE.0?,                       ;NOT - 0?
U 0B71, 0081,2016,0A41,4047,00B6,E      ;24265          NEXT/CS.R0.D.GETS.RB+1.SC.SP    ;
```

```
;24266 .TOC     ''      Character String                : LOCC''
;24267
;24268          ;*****************************************************************
;24269          ;       LOCC
;24270          ;
;24271          ;       INPUT           TEMP5           CHARACTER
;24272          ;                       R0              LENGTH
;24273          ;                       D               ''
;24274          ;                       Q               ORIGINAL PC
;24275          ;                       PC              ADDRESS
;24276          ;                       R1              ''
;24277          ;                       FPDOFFSET       6
;24278          ;
;24279          ;       OUTPUT          R0              #OF BYTE LEFT IN STR
;24280          ;                                       INCL LOCATED BYTE
;24281          ;                       R1              ADD OF BYTE LOCATED
;24282          ;                                       OR ONE BYTE BEYOND
;24283          ;                                       THE STRING
;24284          ;                       CC              NZVC_0100 IF R0=0
;24285          ;                                       NZVC_0000 ELSE
;24286          ;
;24287          ;       RESOURCES       R0              LENGTH
;24288          ;                       R1              ADDRESS
;24289          ;                       Q               ORIGINAL PC
;24290          ;                       D               LENGTH
;24291          ;                       TEMP5           CHARACTER
;24292          ;                       FPDOFFSET       (FOR LOCC-SKPC'S = 6)
;24293          ;                       PC              ADDRESS
;24294          ;                       FLAG0           0
;24295          ;                       FLAG1           0
;24296          ;                       FLAG3           0
;24297          ;*****************************************************************
```

```
                                ;24298          ;*****************************************************************************
                                ;24299          ; Enter at CS.LOC on a WX.EQ.0? split.  =0 is length >0, begin loop to
                                ;24300          ; match characters.  =1 is length =0, exit instruction.
                                ;24301          ; When a match is found, or EOS hit, BUT on IR<2-0> to CS.LOC.MAT.
                                ;24302          ;*****************************************************************************
                                ;24303 =*00
                                ;24304 =*01
                                ;24305 CS.LOC.MAT:
                                ;24306          ;*01--------------------------------;
                                ;24307          M[TEMP10]_Q_D_R[R0],              ;MATCHC[39], is length = 0?
                                ;24308          WX.EQ.0?,
U 0AF5, 0086,A5BE,7A34,0047,08B0,0 322*         ;24309          NEXT/CS.MAT.R0.Q.D
                                ;24310
                                ;24311 CS.LOC:
                                ;24312          ;*10--------------------------------;
                                ;24313          WB_ZEXT(XB)-R[TEMP5] PC_PC+1,    ;Is thie BYTE = character?
U 0AF6, 0081,7014,0B41,6047,08AF,A 385*         ;24314          SIGND CMP?,NEXT/CS.LOC.SPL
                                ;24315
                                ;24316          ;*11--------------------------------;
                                ;24317          PC_Q,CLEAR FPD,                   ;LOCC[3A], finish instruction
U 0AF7, 04E0,003A,403D,8487,0000,2 ;24318       NEXT/GL.NOP.IRD1
                                ;24319
                                ;24320 =10
                                ;24321 CS.LOC.SPL:
                                ;24322          ;10---------------------------------;
                                ;24323          R[R0]_D_RB-1,CCOP1,SIZE[WORD],   ;No match, set CC, is length = 0?
U 0AFA, 0484,0E7D,2A54,0847,08B7,2 372*         ;24324          WX.NE.0?,NEXT/CS.LOC.SPL.1
                                ;24325
                                ;24326          ;11---------------------------------;
                                ;24327          PC_R[R1]_M[PC]-1,IR<2-0>?,        ;We have a match, which inst are we?
U 0AFB, 0C85,AE51,0674,4487,00AF,5 ;24328       NEXT/CS.LOC.MAT
                                ;24329
                                ;24330 =0
                                ;24331 CS.LOC.SPL.1:
                                ;24332          ;0---------------------------------;
                                ;24333          R[R1]_M[PC],IR<2-0>?,CLEAR FPD,  ;Length=0, loop is finished, FPD_0
U 0B72, 04E5,A593,4674,4047,00AF,5 ;24334       NEXT/CS.LOC.MAT                   ;incase we are LOCC, which inst. are we?
                                ;24335
                                ;24336          ;1---------------------------------;
U 0B73, 0880,0036,4B30,18E7,00B7,4 ;24337       .IP.TS?                           ;Length>0, continue with loop after
                                ;24338                                            ;checking for timer service/ints.
                                ;24339
                                ;24340 =0
                                ;24341          ;0---------------------------------;
                                ;24342          WB_ZEXT(XB)-R[TEMP5] PC_PC+1,    ;Nothing pending or timer service,
U 0B74, 0081,7014,0B41,6047,08AF,A 385*         ;24343          SIGND CMP?,NEXT/CS.LOC.SPL        ;compare next BYTE
                                ;24344
                                ;24345          ;1---------------------------------;
                                ;24346          PUSH,INTPEND OR TIMER?,           ;Something pending, RETURN -1 or pack
U 0B75, 0480,0036,4AF0,0047,04F7,0 ;24347       NEXT/IE.SERV.IP.TS2               ;up
```

```
                                ;24348 .TOC     "       Character String                : SKPC"
                                ;24349
                                ;24350          ;*************************************************************************
                                ;24351          ; SKPC uses the same hardware resources and firmware routines as LOCC
                                ;24352          ; except for:
                                ;24353          ;
                                ;24354          ;       SUBROUTINES     CS.C5.S1F.LOP   For source compare
                                ;24355          ;
                                ;24356          ;       OUTPUT          GPR0            #OF BYTE LEFT IN STR
                                ;24357          ;                                       INCL UNEQUAL BYTE
                                ;24358          ;
                                ;24359          ; Enter at CS.SK: on WX.EQ.0? BUT.  =00 is length >0, PUSH to CMPC5
                                ;24360          ; compare routine and RETURN +3.  =01 is length =0, finish inst. now.
                                ;24361          ;*************************************************************************
                                ;24362 =00
                                ;24363 CS.SK:
                                ;24364          ;00-----------------------------;
                                ;24365          WB_D-ZLIT0[4],WB<31-30>?,       ;Is DREG>=4 or <4?
                                ;24366          CLEAR FLAG1,PUSH,
U 0AFC, 0908,0C30,06F0,2047,0CAD,5 377*         ;24367          NEXT/CS.C5.S1F.LOP
                                ;24368
                                ;24369 CS.SK.EXIT:
                                ;24370          ;01-----------------------------;
                                ;24371          PC_Q,CLEAR FPD,                 ;PC_Q, FPD_0, exit inst.
U 0AFD, 04E0,003A,403D,8487,0000,2              ;24372          NEXT/GL.NOP.IRD1                ;FINISHED WITH LOCC
                                ;24373
                                ;24374 =11
                                ;24375          ;11-----------------------------;
                                ;24376          WB_R[R0].CCOP1,SIZE[WORD],      ;Subroutine returns here, set CC,
U 0AFF, 0880,05BE,4014,0847,00AF,D              ;24377          NEXT/CS.SK.EXIT                 ;exit inst.
```

```
                        ;24378 .TOC     "       Character String                : MATCHC"
                        ;24379
                        ;24380          ;*******************************************************************************
                        ;24381          ;       MATCHC
                        ;24382          ;       INPUT           R2              DELTA PC'0'OBJECT LENGTH
                        ;24383          ;                                       (8 BITS)'(8)'(16 BITS)
                        ;24384          ;                       Q
                        ;24385          ;                       R3              OBJECT ADDRESS
                        ;24386          ;                       VA
                        ;24387          ;                       R0              SOURCE LENGTH
                        ;24388          ;                       D
                        ;24389          ;                       PC              SOURCE ADDRESS
                        ;24390          ;                       R1
                        ;24391          ;                       FLAG3           SET (MATCHC FLAG)
                        ;24392          ;
                        ;24393          ;       OUTPUT          R0              #OF UNMAT BYTES OBJ STR
                        ;24394          ;                       R1              ADD NEXT BYTE IN OBJ STR
                        ;24395          ;                                       OR ONE BYTE BEYOND
                        ;24396          ;                       R2              # BYTES LEFT IN SRC STR
                        ;24397          ;                       R3              ADD OF NEXT BYTE BEYOND
                        ;24398          ;                                       MATCH OR SRC STR
                        ;24399          ;                       CC              NZVC_0100 IF R0=0
                        ;24400          ;                                       NZVC_0000 ELSE
                        ;24401          ;
                        ;24402          ;       RESOURCES       R0
                        ;24403          ;                       R1
                        ;24404          ;                       R2
                        ;24405          ;                       R3
                        ;24406          ;                       FPDOFFSET       7
                        ;24407          ;                       VA
                        ;24408          ;                       PC
                        ;24409          ;                       TEMP1
                        ;24410          ;                       TEMP6           CODE
                        ;24411          ;                       Q
                        ;24412          ;                       D
                        ;24413          ;                       MTMP10
                        ;24414          ;                       FLAG0
                        ;24415          ;                       FLAG3           SET FOR MATCHC
                        ;24416          ;
                        ;24417          ;       SUBROUTINES     CS.GEN.RESPC
                        ;24418          ;*******************************************************************************
```

```
                                ;24419 0BA1:
                                ;24420 CS.MAT.SET.FPDOFF.-1:
                                ;24421         ;01*****FORCE ADDRESS**********;
                                ;24422         M[FPDOFFSET]_ZLIT0[7],          ; SRC LEN > OR = OBJ LEN
0BA1, 0586,CC37,0030,3847,00B9,E ;24423         NEXT/CS.MAT.TMP5.-1             ; GO READ 1ST CHAR
                                ;24424
                                ;24425 0BA3:
                                ;24426         ;11*****FORCE ADDRESS**********;
0BA3, 0481,A001,0034,0487,003E,C ;24427         PC_M[PC]+R[R0]                  ; SRC LEN < OBJ LEN
                                ;24428                                         ; GO SET R0-R3 TO PROPER VALUE
                                ;24429                                         ; AND EXIT INSTRUCTION
                                ;24430
                                ;24431 03EC:
                                ;24432         ;********FORCE ADDRESS**********;
03EC, 0485,8592,4034,4047,003E,D ;24433         R[R1]_M[VA]                     ; R1 GETS OBJ ADDRESS
                                ;24434
                                ;24435 03ED:
                                ;24436         ;********FORCE ADDRESS**********;
03ED, 0880,05BE,603D,8047,00B7,8 ;24437         D_R[ZERO]                       ; CLEAR D REG
                                ;24438
                                ;24439 =0
                                ;24440 CS.MAT.EXIT:
                                ;24441         ;0-----------------------------;
                                ;24442         PUSH,R[R3]_M[PC],CLEAR FPD,     ;GPR3_PC, restore PC & RETURN+1
0B78, 0CE5,A592,4034,C047,04AA,5 ;24443         NEXT/CS.GEN.RESPC
                                ;24444
                                ;24445         ;1-----------------------------;
                                ;24446         R[R0]_ZEXT(M[TEMP1]),           ;GPR0_# bytes in the object string
0B7A, 0884,159E,4014,0047,00B9,C ;24447         SIZE[WORD]
                                ;24448
                                ;24449         ;-----------------------------;
0B9E, 0484,05B2,4134,8047,003F,9 ;24450         R[R2]_D,IRD1                    ;GPR2_DREG, **IRD1**
                                ;24451
                                ;24452
                                ;24453
                                ;24454 CS.MAT.TMP5:
                                ;24455         ;-----------------------------;
                                ;24456         R[TEMP5]_ZEXT(M[MDR]),          ;TEMP5_the BYTE, anything pending?
0B9D, 0085,259F,4B01,58E7,00A9,0 ;24457         SIZE[BYTE],IP.TS?
                                ;24458
                                ;24459 =000
                                ;24460         ;000----------------------------;
                                ;24461         PUSH,                           ;RET+4
                                ;24462         M[FPDOFFSET]_ZLIT28[4],         ;FPDOFFSET 40000000
0A90, 0986,CC77,0030,2047,04AB,2 ;24463         NEXT/CS.FIX.MATCHC.IND          ;R2_<31-30>=01(FOR LOCC)
                                ;24464
                                ;24465         ;001----------------------------;
                                ;24466         PUSH,                           ;INTERRUPT
                                ;24467         INTPEND OR TIMER?,              ;BUT INTO IE.SERV ROUTINE
0A91, 0480,0036,4AF0,0047,0417,0 ;24468         NEXT/IE.SERV.IP.TS2             ;
```

```
                                        ;24469 =100
                                        ;24470          ;100-------------------------------;
                                        ;24471          M[FPDOFFSET]_ZLIT0[6],          ;FPDOFFSET GETS AN 6
U 0A94, 0186,CC37,0030,3047,00AF,6      ;24472          NEXT/CS.LOC                     ;LOCC (RET AT CS.MAT.10.D.Q)
                                        ;24473 =
                                        ;24474
                                        ;24475 CS.MAT.TMP5.-1:
                                        ;24476          ;-------------------------------;
U 0B9E, 0986,8EB7,0039,F847,00B9,F      ;24477          M[TEMP8]_OLIT24[13F]            ;TEMP8_3FFFFFFF(FOR AND INTO R2)
                                        ;24478
                                        ;24479          ;-------------------------------;
                                        ;24480          READ,SIZE[BYTE],                ;READ A BYTE
U 0B9F, 0480,0036,4000,0050,00B9,D      ;24481          NEXT/CS.MAT.TMP5                ;MTMP5 GETS 1ST CHAR(READ CHAR)
                                        ;24482 =0
                                        ;24483 CS.MAT.SET.FPDOFF:
                                        ;24484          ;0-------------------------------; OBJECT LENGTH NOT EQUAL TO ZERO
                                        ;24485          WB_D-Q,SIGND CMP?,SIZE[LONG],   ;
U 0B7A, 0C80,0028,0B60,0047,088A,1 368* ;24486          NEXT/CS.MAT.SET.FPDOFF.-1       ; SRC LENGTH < OBJ LENGTH ?
                                        ;24487
                                        ;24488          ;1-------------------------------; OBJECT LENGTH = 0
U 0B7B, 0C85,B592,4034,4047,00B7,8      ;24489          R[R1]_M[VA],NEXT/CS.MAT.EXIT    ; EXIT INSTR AS IF MATCH WAS FOUND
                                        ;24490
                                        ;24491 =00
                                        ;24492 CS.MAT.R0.Q.D:
                                        ;24493          ;00------------------------------;
                                        ;24494          M[TEMP6]_ZLIT8[1],              ;MTMP6 (CODE) GETS A 1
U 0B00, 0986,6DB7,0030,0847,008C,D      ;24495          NEXT/CS.MAT.R0.Q.D.1            ;GET R0,Q,AND D
                                        ;24496
                                        ;24497          ;01------------------------------;
                                        ;24498          PUSH,                           ;RET+1
                                        ;24499          SET FLAG0,                      ;SET FLAG0
                                        ;24500          R[R3]_M[PC],                    ;R3 GETS PC
U 0B01, 0C45,A592,4034,C047,04AA,5      ;24501          NEXT/CS.GEN.RESPC               ;RESTORE PC
                                        ;24502
                                        ;24503
                                        ;24504 CS.MAT.R1.GETS.VA.CL:
                                        ;24505          ;10------------------------------;
                                        ;24506          R[R1]_M[VA],                    ;R1 GETS VA
                                        ;24507          CLEAR FPD,                      ;CLEAR FIRST PART DONE
                                        ;24508          FLAG0?,                         ;
U 0B02, 08E5,B592,4434,4047,00B7,E      ;24509          NEXT/CS.MAT.R2.GETS.10-Q        ;
                                        ;24510 =
                                        ;24511
                                        ;24512 0BCD:
                                        ;24513 CS.MAT.R0.Q.D.1:
                                        ;24514          ;*******FORCE ADDRESS**********;
U 0BCD, 0484,1592,7034,0047,00B7,6      ;24515          R[R0]_Q_D_M[TEMP1]              ;R0,Q,D GET TEMP1(LENGTH)
                                        ;24516
                                        ;24517 0B76:
                                        ;24518          ;********FORCE ADDRESS*********;
                                        ;24519          WB_M[TEMP10]-R[TEMP1],          ;FLAG0_0, is MTEMP10 < length?
U 0B76, 0400,A000,06F0,4047,0849,9 338* ;24520          CLEAR FLAG0,WB<31-30>?
```

```
                                ;24521 =000
                                ;24522 =001
                                ;24523          ;001-------------------------------;
                                ;24524          PUSH,                           ;RET+4
                                ;24525          M[FPDOFFSET] ZLIT28[8],         ;FPDOFFSET_80000000
U 0A99, 0586,CC77,0030,4047,04AB,2 ;24526       NEXT/CS.FIX.MATCHC.IND          ;R2<31-30>_10
                                ;24527
                                ;24528 =011
                                ;24529          ;011-------------------------------;
                                ;24530          PUSH,                           ;RET+1
                                ;24531          M[TEMP2] R[R1],                 ;MTMP2 GETS R1
                                ;24532          SET FLAG0,                      ;SET FLAG0
U 0A9B, 0846,25BE,4034,4047,04AA,5 ;24533       NEXT/CS.GEN.RESPC               ;RESTORE PC
                                ;24534
                                ;24535          ;100-------------------------------;
                                ;24536          R[TEMP2] M[TEMP10]+RB,          ;TEMP2_R1+MTMP10
U 0A9C, 0C84,A001,0030,8047,00B7,D ;24537       NEXT/CS.MAT.R3.GETS.TMP2        ;
                                ;24538
                                ;24539          ;101-------------------------------;
                                ;24540          M[FPDOFFSET] ZLIT0[0],          ;Load FPDOFFSET, set MATCHC FLAG,
U 0A9D, 0D5E,CC37,0030,0047,00A3,9 ;24541       SET FLAG3,NEXT/CS.C3            ;enter CMPC3 code
                                ;24542 =
                                ;24543 =0
                                ;24544 CS.MAT.Q.GETS.Q-R0:
                                ;24545          ;0-------------------------------;
                                ;24546          Q_Q-R[R0],                      ;Q_Q-R0
U 0B7C, 0480,003B,1034,0047,00BA,2 ;24547       NEXT/CS.MAT.R1.GETS.R1-Q        ;
                                ;24548
                                ;24549 CS.MAT.R3.GETS.TMP2:
                                ;24550          ;1-------------------------------;
                                ;24551          R[R3]_M[TEMP2],                 ;R3 GETS PC(TMP2)
U 0B7D, 0484,2592,4034,C047,00B0,2 ;24552       NEXT/CS.MAT.R1.GETS.VA.CL       ;R1 GETS VA CLR FPD CODE
                                ;24553
                                ;24554 =0
                                ;24555 CS.MAT.R2.GETS.10-Q:
                                ;24556          ;0-------------------------------;
                                ;24557          R[R2]_M[TEMP10]-Q,              ;R2 GETS MTMP10-Q
U 0B7E, 0484,A008,0134,8047,003F,9 ;24558       IRD1                            ;FINISHED
```

```
                                ;24559          ;1--------------------------------;
                                ;24560          R[R0]_M[TEMP1],SIZE[LONG],      ;GPR0_TEMP1, set CC, and clear GPR2 &
U 0B7F, 0C84,1592,4024,0847,00B1,B ;24561       CCOP1,NEXT/CS.MTC.R2_0.IRD1     ;IRD1 in MOVTC code
                                ;24562
                                ;24563 CS.MAT.R1.GETS.R1-Q:
                                ;24564          ;--------------------------------;
U 0BA2, 0884,0038,0034,4047,00B7,7 ;24565       R[R1]_RB-Q                      ;R1 GETS R1-Q
                                ;24566
                                ;24567 0B77:
                                ;24568          ;********FORCE ADDRESS**********;
U 0B77, 0C84,0E7C,0034,4487,00BA,4 ;24569       PC_R[R1]_RB+1                   ;R1,PC GETS R1+1
                                ;24570
                                ;24571          ;--------------------------------;
U 0BA4, 0C84,0038,0034,C4A7,00BA,5 ;24572       R[R3]_VA_RB-Q                   ;R3,VA GET R3-Q
                                ;24573
                                ;24574          ;--------------------------------;
                                ;24575          R[R0]_M[TEMP10]-1,              ;R0 GETS MTMP10-1
U 0BA5, 0084,AE51,0034,0047,00BA,1 ;24576       NEXT/CS.MAT.SET.FPDOFF.-1       ;START AGAIN
```

```
;24577 .TOC  ""      Character String                : MOV FPD Pack routine""
;24578
;24579  ;*****************************************************************************
;24580  ;       INPUT           R0              LENGTH
;24581  ;                       R1              SOURCE ADDRESS
;24582  ;                       R2              DELTA PC'0(8BITS'24BITS)
;24583  ;                       R3              DESTINATION ADDRESS
;24584  ;                       PC
;24585  ;                       DELTA PC        FOUND IN R2
;24586  ;                       VA
;24587  ;                       TEMP0<1-0>      number of extra BYTEs to move
;24588  ;                                       after (GPR0 + 4).  This is
;24589  ;                                       relevant only if MTEMP9>0.
;24590  ;                       TEMP5           FILL
;24591  ;                       TEMP6           CODE
;24592  ;                       TEMP7           TABLE ADDRESS
;24593  ;                       MTEMP9          =0 if we were in a standard
;24594  ;                                       move.
;24595  ;                                       >0 if we were MOVC<C5> and were
;24596  ;                                       moving LONGWORDs to an aligned
;24597  ;                                       dest.  GPR0 is adjusted +4
;24598  ;                                       +TEMP0<1-0> due to decrementing
;24599  ;                                       before a WRITE and possible
;24600  ;                                       extra BYTEs.
;24601  ;                       MTMP10          DIFF IN SRC LEN,DES LEN
;24602  ;                       RTMP13          PC
;24603  ;                       FLAGS
;24604  ;
;24605  ;       OUTPUT
;24606  ;
;24607  ; CODES         0,1             2,6             3
;24608  ;-----------------------------------------------------------------------------
;24609  ; R0            PC        /\VA(16)/\R3(16)      PC
;24610  ;     ----------------  ----------------  ----------------
;24611  ; R1            R1              <--             <--
;24612  ;     ----------------  ----------------  ----------------
;24613  ; R2    /\PC(8)S(8)M10(16)      <--             <--
;24614  ;     ----------------  ----------------  ----------------
;24615  ; R3    VA (ALREADY)    PC(IF 2)OR R13(IF 6)  VA (ALREADY)
;24616  ;     ----------------  ----------------  ----------------
;24617  ; R4    R0(16)?(8)C(?)MB?F(8)   <--             <--
;24618  ;     ----------------  ----------------  ----------------
;24619  ; R5    TABLEADDRESS            <--             <--
;24620  ;     ----------------  ----------------  ----------------
;24621  ;-----------------------------------------------------------------------------
;24622  ; /\ := delta     (X) := X bits in length     Y := TEMPY
;24623  ; MY := MTEMPY    MB := MTEMP bit      F := FLAGS    ? := garbage
;24624  ;-----------------------------------------------------------------------------
;24625  ;
;24626  ;*****************************************************************************
```

```
                                ;24627 CS.MOV.PACK.BEGIN:
                                ;24628          ;--------------------------------;
U 0BA6, 0484,7592,4035,4047,00BA,7  ;24629          R[R5]_M[TEMP7]                  ;R5 GETS TABLEADDRESS-PACKING
                                ;24630
                                ;24631          ;--------------------------------;
U 0BA7, 0484,5002,4034,8047,00BA,8  ;24632          R[R2]_M[TEMP5].OR.RB            ;Mask till into GPR2
                                ;24633
                                ;24634          ;--------------------------------;
U 0BA8, 0480,9592,4A70,0047,00B8,0  ;24635          WB_M[TEMP9],WX.NE.0?            ;Are we MOVC<35> moving aligned
                                ;24636                                          ;LONGWORDs?
                                ;24637 =0
                                ;24638 CS.PCK.MOV:
                                ;24639          ;0--------------------------------;
                                ;24640          R[R2].SIZ_M[TEMP10],SIZE[WORD], ;No, GPR2_deltaPC'Fill'MTEMP10
U 0B80, 0082,A592,4014,8047,00BA,B  ;24641          NEXT/CS.PCK.MOV.1
                                ;24642
                                ;24643          ;1--------------------------------;
U 0B81, 0C84,073D,0024,0047,0CBA,9  ;24644          R[R0]_RB+CONX(4)                ;Yes, GPR0_inc. by a LONGWORD
                                ;24645
                                ;24646          ;--------------------------------;
U 0BA9, 0186,0C12,0030,1847,00BA,A  ;24647          M[TEMP0]_MB.AND.ZLITO[3]        ;Mask out <1-0>
                                ;24648
                                ;24649          ;--------------------------------;
                                ;24650          R[R0]_M[TEMP0]+RB,              ;GPR0_RB+0, 1, 2, or 3, adjustment is
U 0BAA, 0484,0001,0034,0047,00B8,0  ;24651          NEXT/CS.PCK.MOV                 ;complete
                                ;24652
                                ;24653 CS.PCK.MOV.1:
                                ;24654          ;--------------------------------;
U 0BAB, 0880,A592,46F0,0047,00B0,5  ;24655          WB_M[TEMP10],WB<31-30>?         ;Is MTEMP10 negative?
                                ;24656
                                ;24657 =01
                                ;24658 CS.MOV.PACK.1:
                                ;24659          ;01-------------------------------;
                                ;24660          M[TEMP7]_FLAGS,                 ;TEMP7_GARBAGE'FLAGS
U 0B05, 0886,7036,4030,0387,00BA,C  ;24661          NEXT/CS.MOV.PACK.2              ;
                                ;24662
                                ;24663          ;11-------------------------------;
                                ;24664          M[TEMP6]_MB.OR.ZLITO[80],       ;MTMP10 IS NEG
U 0B07, 0186,6C12,4034,0047,00B0,5  ;24665          NEXT/CS.MOV.PACK.1              ;OR IN 1 IN MTMP10 BIT
                                ;24666
                                ;24667 CS.MOV.PACK.2:
                                ;24668          ;--------------------------------;
U 0BAC, 0580,7C12,1031,F847,00BA,D  ;24669          Q_M[TEMP7].AND.ZLITO[3F]        ;Q_0'0'0'00FLAGS
                                ;24670
                                ;24671          ;--------------------------------;
U 0BAD, 0084,600A,4035,0047,00BA,E  ;24672          R[R4]_M[TEMP6].OR.Q             ;R4_0'0'CODE'MTMP10 BIT'0'FLAGS
                                ;24673
                                ;24674          ;--------------------------------;
U 0BAE, 0886,82B7,0014,0047,00BA,F  ;24675          M[TEMP8]_R[R0].RR.16            ;TEMP8_WORD'0
                                ;24676
                                ;24677          ;--------------------------------;
U 0BAF, 0084,8002,4035,0047,0081,0  ;24678          R[R4]_M[TEMP8].OR.RB            ;R4_R0'CODE'MTMP10 BIT'?'FLAGS
```

```
                                ;24679 0810:    ;*********FORCE ADDRESS*********;
U 0810, 0D86,6D92,0037,F847,00BB,1              ;24680          M[TEMP6]_MB.AND.ZLIT8[OFF]      ;MTMP6 JUST HAS CODE IN IT
                                ;24681
                                ;24682          ;-------------------------------;
                                ;24683          WB_M[TEMP6]-ZLIT8[3],           ;WBUS_MTMP6-300
U 0BB1, 0980,6D90,0A70,1847,08B8,2 378*         ;24684          WX.NE.0?                        ;NOT EQUAL TO 300?
                                ;24685
                                ;24686 =0
                                ;24687          ;0-------------------------------;
                                ;24688          R[R0]_M[PC],                    ;R0 GETS PC BECAUSE CODE IS 300
U 0B82, 0485,A592,4034,0047,00FE,2              ;24689          NEXT/IE.PACK.DONE               ;GO TO CHARLIE
                                ;24690
                                ;24691          ;1-------------------------------;
                                ;24692          M[TEMP6]_MB.RR.8,               ;MTMP6_CODE SHIFTED TO LEAST SIG BITS
U 0B83, 0086,63B7,0240,0047,08B0,9 337*         ;24693          WB<1-0>?                        ;BIT 1 SET?
                                ;24694
                                ;24695 =01
                                ;24696          ;01------------------------------;
                                ;24697          R[R0]_M[PC],                    ;R0 GETS PC
U 0B09, 0485,A592,4034,0047,00FE,2              ;24698          NEXT/IE.PACK.DONE               ;GO TO CHARLIE
                                ;24699
                                ;24700          ;11------------------------------;
U 0B0B, 0C86,85BE,4034,C047,00BB,2              ;24701          M[TEMP8]_R[R3]                  ;BIT 1 SET CODE =2 OR 6
                                ;24702
                                ;24703          ;-------------------------------;
U 0BB2, 0486,8000,0034,4047,00BB,3              ;24704          M[TEMP8]_MB-R[R1]               ;DELTA R3
                                ;24705
                                ;24706          ;-------------------------------;
                                ;24707          WB_M[TEMP6]-ZLIT0[2],           ;MTMP8-2
U 0BB3, 0980,6C10,0A30,1047,08B8,4 384*         ;24708          WX.EQ.0?                        ;IS IT A 2?
                                ;24709
                                ;24710 =0
                                ;24711          ;0-------------------------------;
U 0B84, 0880,05BE,4033,4487,00B8,5              ;24712          PC_R[TEMP13]                    ;CODE=6
                                ;24713
                                ;24714          ;1-------------------------------;
U 0B85, 0885,A592,5034,C047,00BB,4              ;24715          R[R3]_Q_M[PC]                   ;CODE = 2
                                ;24716
                                ;24717          ;-------------------------------;
U 0BB4, 0084,4008,0034,0047,00BB,5              ;24718          R[R0]_M[TEMP4]-Q                ;RTMP13_VA-PC(OR RTMP13)
                                ;24719
                                ;24720          ;-------------------------------;
                                ;24721          R[R0]_M[TEMP8].OR.(RB.RR.16),   ;R0_/\VA'/\R3
U 0BB5, 0484,8292,4014,0047,00FE,2              ;24722          NEXT/IE.PACK.DONE               ;GO TO CHARLIE
```

```
                                ;24723 .TOC     "       Character String                : MOV FPD Unpack routine"
                                ;24724
                                ;24725          ;*************************************************************************
                                ;24726          ; The input of the unpack code is the output of the pack code.
                                ;24727          ; The output of the unpack code is the input of the pack code.
                                ;24728          ; Refer to the pack routine banner a few pages back.
                                ;24729          ;*************************************************************************
                                ;24730
                                ;24731 .REGION/IRD1.R1L,IRD1.R1H
                                ;24732 =000
                                ;24733 CS.MOV.UNPACK:
                                ;24734          ;000-------------------------------;
U 0328, 0486,75BE,4035,4047,00BB,6  ;24735      M[TEMP7]_R[R5]                  ;TEMP7 GETS TABLEADDRESS
                                ;24736 =
                                ;24737
                                ;24738 .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                                ;24739
                                ;24740          ;-------------------------------;
U 0BB6, 0886,C5B7,0030,0047,00BB,7  ;24741      M[FPDOFFSET]_0                  ;Load FPDOFFSET
                                ;24742
                                ;24743          ;-------------------------------;
U 0BB7, 0886,95B7,0030,0047,00BB,8  ;24744      M[TEMP9]_0                      ;Zero GPR0 decrement flag
                                ;24745
                                ;24746          ;-------------------------------;
U 0BB8, 0086,52B7,0014,8047,00B8,6  ;24747      M[TEMP5]_R[R2].RR.16            ;MTMP5 GETS R2 ROT BY 16
                                ;24748
                                ;24749 =0
                                ;24750          ;0--------------------------------;
                                ;24751          PUSH,                           ;RET+1
                                ;24752          M[TEMP5]_ZEXT(MB),              ;EXP FILLER BUT 1ST 0'FILL
                                ;24753          SIZE[BYTE],                     ;BYTE SIZE
U 0B96, 0886,559E,4000,0047,04A9,A  ;24754      NEXT/CS.EXP.FILLER              ;EXP FILLER
                                ;24755
                                ;24756          ;1--------------------------------;
U 0B87, 0486,A5BE,4034,8047,00BB,9  ;24757      M[TEMP10]_R[R2]                 ;TEMP10_GARBAGE'TEMP10
                                ;24758
                                ;24759          ;-------------------------------;
U 0BB9, 0D86,2C37,0034,0047,00BB,A  ;24760      M[TEMP2]_ZLIT0[80]              ;FIND OUT IF MTMP10 IS NEGATED
                                ;24761
                                ;24762          ;-------------------------------;
                                ;24763          WB_M[TEMP2].AND.R[R4],          ;IS BIT SET?
U 0BBA, 0C80,2002,0A75,0047,00B0,C  ;24764      WX.NE.0?                        ;
                                ;24765
                                ;24766 =00
                                ;24767          ;00-------------------------------;
                                ;24768          M[TEMP10]_ZEXT(MB),             ;MTMP10 0'MTMP10(POSITIVE)
                                ;24769          SIZE[WORD],                     ;WORD EXTENDED
U 0B0C, 0486,A59E,4010,0047,00B0,E  ;24770      NEXT/CS.MOV.UNPACK.1            ;
                                ;24771
                                ;24772          ;01-------------------------------;
                                ;24773          PUSH,                           ;RET+1
                                ;24774          M[TEMP2]_-1,                    ;BIT IS SET
U 0B0D, 0886,2E77,0030,0047,04BE,A  ;24775      NEXT/CS.ONP.TMP10.IS.NEG        ;ADJUST MTMP10
```

```
                                        ;24776 CS.MOV.UNPACK.1:
                                        ;24777          ;10------------------------------;
U 0B0E, 0080,05BE,6035,0307,00BB,B      ;24778          FLAGS_D_R[R4]                   ;FLAGS ZRE RESTORES D_R4
                                        ;24779 =
                                        ;24780
                                        ;24781          ;------------------------------;
U 0BBB, 0186,6DB2,0030,7847,00BB,C      ;24782          M[TEMP6]_D.AND.ZLIT8[0F]        ;TEMP6 GETS CODE RESTORED
                                        ;24783
                                        ;24784          ;------------------------------;
U 0BBC, 0486,82B7,0015,0047,00BB,0      ;24785          M[TEMP8]_R[R4].RR.16            ;MTMP8 GETS GARBAGE'R0
                                        ;24786
                                        ;24787 0BB0:
                                        ;24788          ;*******FORCE ADDRESS***********;
U 0BB0, 0486,15BE,4034,0487,00BB,E      ;24789          PC_M[TEMP1]_R[R0]               ;PC(TMP1) MAY BE RESTORED
                                        ;24790
                                        ;24791          ;------------------------------;
                                        ;24792          R[R0]_D_ZEXT(M[TEMP8]),         ;GPR0 & DREG are restored
U 0BBE, 0484,859E,6014,0047,00BB,F      ;24793          SIZE[WORD]
                                        ;24794
                                        ;24795          ;------------------------------;
                                        ;24796          WB_M[TEMP6]-ZLIT8[3],           ;IS CODE = 300?
U 0BBF, 0D80,6D90,0A30,1847,08B8,8 384* ;24797          WX.EQ.0?                        ;
                                        ;24798
                                        ;24799 =0
                                        ;24800          ;0------------------------------;
                                        ;24801          WB_M[TEMP6]-ZLIT8[2],           ;No, is code < 200 (codes 000 or 100)?
U 0B88, 0980,6D90,06F0,1047,08B1,1 377* ;24802          WB<31-30>?,NEXT/CS.MOV.UNPACK.3
                                        ;24803
                                        ;24804          ;1------------------------------;
U 0B89, 0586,CC37,0030,4847,00BC,0      ;24805          M[FPDOFFSET]_ZLIT0[9.]          ;Yes, reenter MOVC5 seperate flows
                                        ;24806
                                        ;24807          ;------------------------------;
                                        ;24808          VA_M[TEMP4]_R[R3],              ;Is address LONGWORD aligned?
                                        ;24809          WB<1-0>.NE.0?,
U 0BC0, 0886,45BE,43B4,C4A7,00B4,4      ;24810          NEXT/CS.M5.WR
                                        ;24811
                                        ;24812 =01
                                        ;24813 CS.MOV.UNPACK.3:
                                        ;24814          ;01------------------------------;
                                        ;24815          R[TEMP3]_M[PC].RR.16,           ;MTMP6(CODE) > OR = 200
U 0B11, 0485,A3B7,0010,C047,00BC,1      ;24816          NEXT/CS.MOV.UNPACK.4            ;TEMP3_GARBAGE'/\VA
                                        ;24817
                                        ;24818          ;11------------------------------;
                                        ;24819          VA_M[TEMP4]_R[R3],              ;Are we LONGWORD aligned?
                                        ;24820          WB<1-0>.NE.0?,
U 0B13, 0086,45BE,43B4,C4A7,00A5,0      ;24821          NEXT/CS.M3.WR
```

```
                                        ;24822 CS.MOV.UNPACK.4:
                                        ;24823          ;--------------------------------;
                                        ;24824          M[TEMP3]_ZEXT(MB),              ;MTMP3_0'/\VA
U 0BC1, 0486,359E,4010,0047,00BC,2      ;24825          SIZE[WORD]                      ;ZERO EXT BY A WORD
                                        ;24826
                                        ;24827          ;--------------------------------;
U 0BC2, 0C86,45BE,4034,C047,00BC,3      ;24828          M[TEMP4]_R[R3]                  ;TEMP4 GETS R3
                                        ;24829
                                        ;24830          ;--------------------------------;
U 0BC3, 0480,3001,1034,C4A7,00BC,4      ;24831          VA_Q_M[TEMP3]+R[R3]             ;VA GETS RESTORED
                                        ;24832
                                        ;24833          ;--------------------------------;
                                        ;24834          WB_M[TEMP6]-ZLIT8[2],           ;IS CODE = 2?
U 0BC4, 0980,6D90,0A30,1047,08B1,4 384* ;24835          WX.EQ.0?                        ;
                                        ;24836
                                        ;24837 =00
                                        ;24838          ;00--------------------------------;
U 0B14, 0880,0036,44F0,0047,00B1,8      ;24839          FLAG3?,NEXT/CS.UNP.MOV          ;Are we writing backwards?
                                        ;24840
                                        ;24841          ;01--------------------------------;
                                        ;24842          PUSH,                           ;RET+1
                                        ;24843          PC_D_M[TEMP4],                  ;PC GETS FORMER R3
U 0B15, 0080,4592,6030,0487,04BC,7      ;24844          NEXT7CS.UNP.GET.R3              ;GET R3 FROM DELTA R3
                                        ;24845
                                        ;24846          ;10--------------------------------;
U 0B16, 0086,403A,403D,8047,00A4,8      ;24847          M[TEMP4]_Q,NEXT/CS.M3.WR.BACK   ;TEMP4_QREG (copy of VA), reenter in
                                        ;24848                                          ;writing backward code of MOVC3
                                        ;24849 =
                                        ;24850
                                        ;24851 =00
                                        ;24852 CS.UNP.MOV:
                                        ;24853          ;00--------------------------------;
                                        ;24854          VA_R[R3],                       ; VA GETS UNMODIFIED R3
U 0B18, 0880,05BE,4034,C4A7,00BC,6      ;24855          NEXT/CS.UNP.MOV.1               ;
                                        ;24856
                                        ;24857          ;01--------------------------------;
                                        ;24858          PUSH,                           ;RET+1
                                        ;24859          PC_R[TEMP13]_M[TEMP4],          ;RTMP13,PC GET TMP4
U 0B19, 0484,4592,4033,4487,04BC,7      ;24860          NEXT/CS.UNP.GET.R3              ;
                                        ;24861
                                        ;24862          ;10--------------------------------;
U 0B1A, 0885,B592,4031,0047,00BC,5      ;24863          R[TEMP4]_M[VA]                  ;TEMP4_VA, reenter inst.
                                        ;24864 =
                                        ;24865
                                        ;24866 CS.UNP.MOV.MTC:
                                        ;24867          ;--------------------------------;
                                        ;24868          M[FPDOFFSET]_ZLIT0[9.],         ;Pack up instead of RETURN+4
U 0BC5, 0586,CC37,0030,4847,00AB,C      ;24869          NEXT/CS.MTC.QR
                                        ;24870
                                        ;24871 CS.UNP.MOV.1:
                                        ;24872          ;--------------------------------;
U 0BC6, 0480,05BE,4034,4487,00BC,5      ;24873          PC_R[R1],NEXT/CS.UNP.MOV.MTC    ;PC_GPR1, reenter inst.
```

```
                                    ;24874 CS.UNP.GET.R3:
                                    ;24875          ;--------------------------------;
                                    ;24876          R[TEMP3] ZEXT(M[TEMP1]),        ;TMP3_0'TMP1(R0 OR /\R3)
U 0BC7, 0484,159E,4010,C047,00BC,8  ;24877          SIZE[WORD]                      ;ZERO EXTEND BY A WORD
                                    ;24878
                                    ;24879          ;--------------------------------;
U 0BC8, 0C86,3001,0034,4047,00BC,9  ;24880          M[TEMP3]_MB+R[R1]               ;TMP3_R1+/\R3
                                    ;24881
                                    ;24882          ;--------------------------------;
                                    ;24883          R[R3] M[TEMP3],                 ;R3 RESTORED
U 0BC9, 0C84,3592,40B4,C047,0000,1  ;24884          RETURN [1]                      ;RE1+1
```

```
;24885 .TOC    ''      Character String                : CMPC FPD Pack routine''
;24886
;24887 ;*************************************************************************
;24888 ;       CMPC    INPUT                   R0              LENGTH
;24889 ;                                       R1              SOURCE ADDRESS
;24890 ;                                       R2              DELTA PC'S(8BITS'24BITS)
;24891 ;                                       R3              DESTINATION ADDRESS
;24892 ;                                       D               LENGTH
;24893 ;                                       Q               VALUE FOR MATCHC
;24894 ;                                       PC
;24895 ;                                       VA
;24896 ;                                       FLAGS
;24897 ;                                       TEMP5           FILL
;24898 ;                                       TEMP6           CODE
;24899 ;                                       MTMP10          LENGTH(DIFF IN LENS)
;24900 ;                                       DELTA PC        SEE R2
;24901 ;
;24902 ;               RESOURCES               TEMP2
;24903 ;
;24904 ;               OUTPUT
;24905 ;
;24906 ; CODES         0,1             2,3             4,5
;24907 ;-------------------------------------------------------------------------
;24908 ; R0    M10(16)R0(16)           <--         D OR MTMP10
;24909 ;       ---------------_---------------_---------------_
;24910 ; R1            PC              <--             R1
;24911 ;       ---------------_---------------_---------------_
;24912 ; R2    /\PC(8)5(8)C(8)MB?F(6)  <--             <--
;24913 ;       */\PC(8)Q(16)MB?F(6)
;24914 ;       ---------------_---------------_---------------_
;24915 ; R3    VA (ALREADY)            R3              PC
;24916 ;       ---------------_---------------_---------------_
;24917 ;-------------------------------------------------------------------------
;24918 ; /\ MEANS DELTA        (16) MEANS 16 BITS LONG          5 MEANS TEMP5(M8 MTMP8)
;24919 ; MB MEANS MTMP BIT     F MEANS FLAGS                    ? MEANS UNKNOWN QUANTITY
;24920 ; * WHAT R2 CONTAINS IF MATCHC IS OPCODE AND THIS MAY HAPPEN WHEN CODE =0 OR 1
;24921 ;-------------------------------------------------------------------------
;24922 ;*************************************************************************
```

```
                                ;24923 CS.CMP.PACK.BEGIN:
                                ;24924          ;--------------------------------;
U 0BCA, 0880,A592,46F0,0047,00B1,D  ;24925      WB_M[TEMP10],WB<31-30>?          ;Is MTEMP10 negative?
                                ;24926
                                ;24927 =01
                                ;24928          ;01-------------------------------;
                                ;24929          M[TEMP2]_MB.AND.ZLIT0[3F],       ;TEMP2_0'FLAGS BIT FOR MTMP10 IS NOT SET
U 0B1D, 0186,2C12,0031,F847,00B8,A  ;24930      NEXT/CS.CMP.PACK.1               ;
                                ;24931
                                ;24932          ;11-------------------------------;
U 0B1F, 0D86,2C12,4034,0047,00BC,B  ;24933      M[TEMP2]_MB.OR.ZLIT0[80]         ;TEMP2_GARBAGE(24)'1'?'FLAGS
                                ;24934
                                ;24935          ;--------------------------------;
                                ;24936          M[TEMP2]_ZEXT(MB),               ;NOW EXT BY A BYTE
                                ;24937          SIZE[BYTE],                      ;ZERO EXTEND BY A BYTE
U 0BCB, 0486,259E,4000,0047,00B8,A  ;24938      NEXT/CS.CMP.PACK.1               ;R2_0'1'?'FLAGS
                                ;24939
                                ;24940 =0
                                ;24941 CS.CMP.PACK.1:
                                ;24942          ;0--------------------------------;
                                ;24943          PUSH,                            ;RET+1
                                ;24944          M[TEMP2]_MB.OR.R[TEMP6],         ;TMP2_0'CODE'BIT10'?'FLAGS
U 0B8A, 0C86,2002,4031,8047,04A9,A  ;24945      NEXT/CS.EXP.FILLER               ;EXPAND FILLER
                                ;24946
                                ;24947          ;1--------------------------------;
                                ;24948          R[R2].SIZ_M[TEMP2],              ;R2_R2'TMP2(WORD)
U 0B8B, 0C82,2592,4014,8047,00BC,  ;24949       SIZE[WORD]                       ;
                                ;24950
                                ;24951          ;--------------------------------;
U 0BCC, 0986,5D12,0037,F847,0081,2  ;24952      M[TEMP5]_MB.AND.ZLIT16[0FF]      ;TEMP5_0'FILL'0'0
                                ;24953
                                ;24954 0812:
                                ;24955          ;********FORCE ADDRESS***********;
J 0812, 0484,5002,4034,8047,00BC,E  ;24956      R[R2]_M[TEMP5].OR.RB             ;R2_R2'FILL'R2(WORD)
                                ;24957
                                ;24958          ;--------------------------------;
U 0BCE, 0580,6D90,06F0,2047,08AA,1 377*  ;24959  WB_M[TEMP6]-ZLIT8[4],WB<31-30>?  ;Is TEMP6 >=400 or <400?
```

```
                        ;24960 =000
                        ;24961 =001
                        ;24962          ;001-------------------------------;
0AA1, 0485,A592,4034,C047,00FE,2        ;24963          R[R3]_M[PC],NEXT/IE.PACK.DONE   ;TEMP6>=400, finish packing
                        ;24964
                        ;24965 =011
                        ;24966          ;011-------------------------------;
                        ;24967          PUSH,                           ;RET+1
                        ;24968          SIZE[WORD],                     ;SIZE WORD
                        ;24969          R[TEMP6].SIZ_M[TEMP10],         ;TEMP6_0'MTMP10(WORD)
0AA3, 0882,A592,4011,8047,04BE,9        ;24970          NEXT/CS.R1.GETS.PC              ;
                        ;24971
                        ;24972          ;100-------------------------------;
                        ;24973          M[TEMP6]_MB.RR.16,              ;TMP6_MTMP10'0(WORD)
0AA4, 0086,6587,0400,0047,00AA,6        ;24974          FLAG3?                          ;MATCRC?
                        ;24975
                        ;24976 =0
                        ;24977 CS.CMP.PACK.2:
                        ;24978          ;0---------------------------------;
                        ;24979          R[R0]_M[TEMP6].OR.RB,           ;R0_MTMP10'R0
0AA6, 0084,6002,4034,0047,00FF,2        ;24980          NEXT/IE.PACK.DONE               ;GO TO CHARLIE
                        ;24981
                        ;24982          ;1---------------------------------;
                        ;24983          R[R2]_RB.RR.8,                  ;R2 GETS ROTATED
0AA7, 0484,02B7,0404,8047,00A8,0        ;24984          FLAG0?                          ;IS Q =TMP10
                        ;24985
                        ;24986 =0
                        ;24987          ;0---------------------------------;
                        ;24988          M[TEMP2]_Q,                     ;MTMP2_Q FOR MATCHC
0BBC, 0886,2058,4030,8047,00BC,F        ;24989          NEXT/CS.CMP.PACK.3              ;
                        ;24990
                        ;24991          ;1---------------------------------;
                        ;24992          M[TEMP2]_R[TEMP1],              ;WHAT WE WANT TO SAVE IS TMP1
0BBD, 0086,25BE,4030,8047,00BC,F        ;24993          NEXT/CS.CMP.PACK.3              ;
                        ;24994
                        ;24995 CS.CMP.PACK.3:
                        ;24996          ;---------------------------------;
                        ;24997          R[R2].SIZ_M[TEMP2],             ;R2_R2'TEMP2(WORD)
0BBE, 0482,2593,4034,8047,00BD,0        ;24998          SIZE[WORD]                      ;
                        ;24999
                        ;25000          ;---------------------------------;
                        ;25001          R[R2]_RB.RL.8,                  ;R2 GETS ROTATED BACK
0BBF, 0484,02B7,0024,8047,00AA,6        ;25002          NEXT/CS.CMP.PACK.2              ;R2_R2'Q(16)'R2
```

```
                        ;25003 .TOC     "       Character String                : CMPC FPD Unpack routine"
                        ;25004
                        ;25005          ;*************************************************************************
                        ;25006          ; The input of the unpack code is the output of the pack code.
                        ;25007          ; The output of the unpack code is the input of the pack code.
                        ;25008          ; Refer to the pack routine banner a few pages back.
                        ;25009          ;*************************************************************************
                        ;25010
                        ;25011 .REGION/IRD1.R1L,IRD1.R1H
                        ;25012 =000
                        ;25013 CS.CMP.UNPACK:
                        ;25014          ;000-----------------------------;
0330, 0086,65BE,5034,8307,00BD,1        ;25015          M[TEMP6]_Q_FLAGS_R[R2]
                        ;25016 =
                        ;25017
                        ;25018 .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                        ;25019
                        ;25020 CS.CMP.UNPACK.BEGIN:
                        ;25021          ;-------------------------------;
08D1, 0886,C5B7,0030,0047,00BD,2        ;25022          M[FPDOFFSET]_0                  ;Load FPDOFFSET
                        ;25023
                        ;25024          ;-------------------------------;
08D2, 0586,6D92,0030,7847,0082,0        ;25025          M[TEMP6]_MB.AND.ZLIT8[0F]       ;MTMP6 GETS CODE
                        ;25026
                        ;25027 =00
                        ;25028          ;00-----------------------------;
                        ;25029          PUSH,                           ;RET+1
                        ;25030          MDR_M[TEMP5]_R[R2].RR.16,       ;MDR,TEMP5 GETS R2 ROTATED BY A WORD
0820, 0086,52B7,0014,8467,04BF,8        ;25031          NEXT/CS.PC.GETS.R1              ;PC_R1
                        ;25032
                        ;25033          ;01-----------------------------;
                        ;25034          PUSH,                           ;RET+1
                        ;25035          M[TEMP5]_ZEXT(MB),              ;TMP5_0'FILLER(BYTE)
                        ;25036          SIZE[BYTE],                     ;BYTE ZERO EXTEND
0821, 0886,559E,4000,0047,0449,A        ;25037          NEXT/CS.EXP.FILLER              ;THEN EXPAND IT
                        ;25038
                        ;25039          ;10-----------------------------;
                        ;25040          M[TEMP2]_ZLIT0[80],             ;TEMP2_80
0822, 0186,2C37,04F4,0047,0088,E        ;25041          FLAG3?
                        ;25042 =
```

```
                                        ;25043 =0
                                        ;25044          ;0--------------------------------;
                                        ;25045          WB_M[T[MP6]-ZLIT8[4],           ;Is TEMP6>=400 or <400?
                                        ;25046          WB<31-30>?,
U 0B8E, 0D80,6D90,06F0,2047,08B2,5 377* ;25047          NEXT/CS.CMP.UNP.1A
                                        ;25048
                                        ;25049          ;1--------------------------------;
                                        ;25050          M[TEMP6]_ZLIT8[1],              ;MATCHC CODE=1
U 0B8F, 0186,6DB7,0030,0847,00B2,7      ;25051          NEXT/CS.CMP.UNP.1B              ;
                                        ;25052
                                        ;25053 =01
                                        ;25054 CS.CMP.UNP.1A:
                                        ;25055          ;01-------------------------------;
                                        ;25056          M[TEMP10]_D_RLR0],              ;CODE = 400 OR 500
U 0B25, 0C86,A5BE,6034,0047,00BD,7      ;25057          NEXT/CS.CMP.UNPACK.3            ;
                                        ;25058
                                        ;25059 CS.CMP.UNP.1B:
                                        ;25060          ;11-------------------------------;
U 0B27, 0486,A2B7,0014,0047,00BD,3      ;25061          M[TEMP10]_R[R0].RR.16           ;MTMP10 GETS IT'S LEST SIG BITS
                                        ;25062
                                        ;25063          ;--------------------------------;
                                        ;25064          WB_M[TEMP2].AND.R[R2],          ;IS MTMP10 NEG?
U 0BD3, 0C80,2002,0A74,8047,00B2,8      ;25065          WX.NE.0?                        ;
                                        ;25066
                                        ;25067 =00
                                        ;25068          ;00-------------------------------;
                                        ;25069          M[TEMP10]_ZEXT(MB),             ;MTMP10 IS POS
                                        ;25070          SIZE[WORD],                     ;ZERO EXT BY A WORD
U 0B28, 0486,A59E,4010,0047,00B2,A      ;25071          NEXT/CS.CMP.UNPACK.4            ;GO TO FLOW
                                        ;25072
                                        ;25073          ;01-------------------------------;
                                        ;25074          PUSH,                           ;RET+1
                                        ;25075          M[TEMP2]_-1,                    ;MTMP2 IS NEG,MTMP2_FFFFFFFF
U 0B29, 0886,2E77,0030,0047,04BF,A      ;25076          NEXT/CS.UNP.TMP10.IS.NEG        ;MAKE MTMP10_FFFF'MTMP10
                                        ;25077
                                        ;25078 CS.CMP.UNPACK.4:
                                        ;25079          ;10-------------------------------;
U 0B2A, 0C86,25BE,4034,0047,00BD,4      ;25080          M[TEMP2]_R[R0]                  ;TMP2 GETS R0
                                        ;25081 =
                                        ;25082
                                        ;25083          ;--------------------------------;
                                        ;25084          R[R0]_D_ZEXT(M[TEMP2]),         ;R0.D GET 0'R0(WORD)
                                        ;25085          SIZE[WORD],                     ;ZERO EXTEND A WORD
U 0BD4, 0884,259E,64D4,0047,00B9,0      ;25086          FLAG3?                          ;MATCHC?
```

```
                                ;25087 =0
                                ;25088 CS.CMP.UNPACK.6:
                                ;25089          ;0-------------------------------;
                                ;25090          WB_M[TEMP6]-ZLIT8[2],           ;Is TEMP6 >=200 or <200?
                                ;25091          WB<31-30>?,                     ;(to VA_GPR3 code)
U 0B90, 0980,6D90,06F0,1047,08B2,D 377*  ;25092          NEXT/CS.CMP.UNPACK.5
                                ;25093
                                ;25094          ;1-------------------------------;
U 0B91, 0481,23B7,0020,0467,00BD,5       ;25095          MDR_M[MDR].RL.8                 ;MDR_MDR ROTATED 8 BITS TO THE LEFT
                                ;25096
                                ;25097          ;-------------------------------;
                                ;25098          R[TEMP1]_Q_ZEXT(M[MDR]),        ;Q_OLD VALUE BACK(MATCHC CODE)
U 0BD5, 0085,259E,5010,4047,00BD,6       ;25099          SIZE[WORD]                      ;Q_0'Q(WORD)
                                ;25100
                                ;25101          ;-------------------------------;
                                ;25102          R[R2].SIZ_M[TEMP1],             ;R2_/\PC'?'OLD R2
                                ;25103          SIZE[WORD],                     ;Q_0'Q(WORD)
U 0BD6, 0C82,1592,4014,8047,00B9,0       ;25104          NEXT/CS.CMP.UNPACK.6            ;
                                ;25105
                                ;25106 =01
                                ;25107 CS.CMP.UNPACK.5:
                                ;25108          ;01------------------------------;
                                ;25109          M[FPDOFFSET]_ZLIT0[8.],         ;Reenter CMPC5, Source 1 vs. FILL
U 0B2D, 0986,CC37,0030,4047,00BD,9       ;25110          NEXT/CS.UNP.CMP.C5S1
                                ;25111
                                ;25112          ;11------------------------------;
                                ;25113          VA_R[R3],WB<1-0>.NE.0?,         ;VA GPR3, is address LONGWORD aligned?
U 0B2F, 0C80,05BE,43B4,C4A7,00A7,0       ;25114          NEXT/CS.C3.LOP
                                ;25115
                                ;25116
                                ;25117 CS.CMP.UNPACK.3:
                                ;25118          ;-------------------------------;
U 0BD7, 0186,CC37,0030,4047,00BD,8       ;25119          M[FPDOFFSET]_ZLIT0[8]
                                ;25120
                                ;25121          ;-------------------------------;
                                ;25122          PC_R[R3],                       ;PC GETS R3
U 0BD8, 0080,05BE,4034,C487,00B9,9       ;25123          NEXT/CS.C5.S2F.1
                                ;25124
                                ;25125
                                ;25126 CS.UNP.CMP.C5S1:
                                ;25127          ;-------------------------------;
U 0BD9, 0080,05BF,4034,C4A7,00A8,8       ;25128          VA_R[R3],NEXT/CS.C5             ;VA_GPR3
```

```
                                ;25129 .TOC     "       Character String                : SCANC and SPANC FPD Pack routine"
                                ;25130
                                ;25131          ;*************************************************************************
                                ;25132          ;       INPUT           R0              LENGTH
                                ;25133          ;                       R2              ORIGINAL PC
                                ;25134          ;                       R3              R3
                                ;25135          ;                       TEMP5           FILL
                                ;25136          ;                       D               SAME AS R0
                                ;25137          ;                       PC              MUST PUT INTO R1
                                ;25138          ;                       FLAGS
                                ;25139          ;
                                ;25140          ;       OUTPUT
                                ;25141          ;
                                ;25142          ;-------------------------------------------------------------------------
                                ;25143          ; R0            R0(16)    TEMP5(8) ??FLAGS(6)
                                ;25144          ;       --------_-------- --------_--------
                                ;25145          ; R1                    PC(32)
                                ;25146          ;       -------- -------- -------- --------
                                ;25147          ; R2            R2(ORIGINAL PC)(32)
                                ;25148          ;       --------_-------- --------_--------
                                ;25149          ; R3                    R3(32)
                                ;25150          ;       --------_--------_--------_--------
                                ;25151          ;-------------------------------------------------------------------------
                                ;25152          ; (16) MEANS 16 BITS LONG       ? MEANS UNKNOWN QUANTITY
                                ;25153          ;-------------------------------------------------------------------------
                                ;25154          ;
                                ;25155          ;*************************************************************************
                                ;25156
                                ;25157 =0
                                ;25158 CS.SC.SPAN.PACK.BEGI:
                                ;25159                  ;0-------------------------------;
                                ;25160                  PUSH,                           ;RET+1
                                ;25161                  R[R0]_RB.RL.16,                 ;R0 GETS ROTATED BY A WORD
U 0B92, 0884,02B7,0014,0047,04A9,A      ;25162          NEXT/CS.EXP.FILLER              ;EXP FILLER
                                ;25163
                                ;25164                  ;1-------------------------------;
U 0B93, 0586,5D92,0057,F847,00BD,A      ;25165          M[TEMP5]_MB.AND.ZLIT8[OFF]      ;TMP5_0'0'FILL'0
                                ;25166
                                ;25167                  ;-------------------------------;
U 0BDA, 0884,5002,4034,0047,00BD,B      ;25168          R[R0]_M[TEMP5].OR.RB            ;R0_R0'TMP5'0
                                ;25169
                                ;25170                  ;-------------------------------;
                                ;25171          R[R0].SIZ_FLAGS,                        ;R0_R0'TMP5'??FLAGS
                                ;25172          SIZE[BYTE],                             ;ONLY WRITE A BYTE
U 0BDB, 0C82,0036,4484,0387,00AA,8      ;25173          FLAG2?                          ;HAS PC BEEN INCREMENTED?
                                ;25174
                                ;25175 =0**
                                ;25176                  ;0**----------------------------;
                                ;25177          R[R1]_M[PC],                            ;NOT INCREMENTED SO JUST STORE PC
U 0AA8, 0085,A592,4034,4047,00FE,2      ;25178          NEXT/TE.PACK.DONE               ;GO TO CHARLIE
                                ;25179
                                ;25180 =1**
                                ;25181                  ;1**----------------------------;
                                ;25182          R[R1]_M[PC]-1,                          ;INCREMENTED SO STORE PC-1
U 0AAC, 0085,AE51,0034,4047,00FE,2      ;25183          NEXT/TE.PACK.DONE               ;GO TO CHARLIE
```

```
                                ;25184 .TOC     "       Character String                : SCANC and SPANC FPD Unpack routine
                                ;25185
                                ;25186          ;*************************************************************************
                                ;25187          ; The input of the unpack code is the output of the pack code.
                                ;25188          ; The output of the unpack code is the input of the pack code.
                                ;25189          ; Refer to the pack routine banner a few pages back.
                                ;25190          ;*************************************************************************
                                ;25191
                                ;25192 .REGION/IRD1.R1L,IRD1.R1H
                                ;25193 =000
                                ;25194 CS.SC.SPAN.UNPACK:
                                ;25195          ;000-------------------------------;
U 0338, 0086,52B7,0004,0047,00B9,4  ;25196      M[TEMP5]_R[R0].RR.8             ;MTMP5_GARBAGE'FILL
                                ;25197 =
                                ;25198
                                ;25199 .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                                ;25200 =0
                                ;25201          ;0-------------------------------;
                                ;25202          PUSH,                           ;RET+1
                                ;25203          M[TEMP5] MB.AND.ZLIT0[OFF],     ;TEMP5 0'FILL
U 0B94, 0586,5C12,0037,F847,04BE,8  ;25204      NEXT/CS.PC.GETS.R1              ;PC GETS R1 ONE WORD SUBROUTINE
                                ;25205
                                ;25206          ;1-------------------------------;
U 0B95, 0880,05BE,4034,0307,00BD,C  ;25207      FLAGS_R[R0]                     ;FLAGS GET LEAST SIG 6 BITS OF R0
                                ;25208
                                ;25209          ;-------------------------------;
U 0BDC, 0C86,A2B7,0014,0047,00BD,D  ;25210      M[TEMP10]_R[R0].RR.16           ;MTMP10_R0 ROTATED BY A WORD
                                ;25211
                                ;25212          ;-------------------------------;
                                ;25213          R[R0] D ZEXT(M[TEMP10]),        ;R0,D 0'LEFT 16 BITS OF R0
                                ;25214          SIZE[WORD],                     ;ZERO~EXTEND BY A WORD
U 0BDD, 0084,A59E,6014,0047,00AD,8  ;25215      NEXT/CS.OS.SCSP.BEG             ;SCANAC SPANC BEGINS
```

```
                    :25216 .TOC     "       Character String                : LOCC and SKPC FPD Pack routine"
                    :25217
                    :25218          ;*************************************************************************
                    :25219          ;       INPUT           R0              LENGTH
                    :25220          ;                       R1              SHOULD GET PC
                    :25221          ;                       TEMP5           CHAR
                    :25222          ;                       D               SAME AS R0
                    :25223          ;                       PC              MUST PUT INTO R1
                    :25224          ;                       Q               ORIGINAL PC
                    :25225          ;
                    :25226          ;       RESOURCES       TEMP1
                    :25227          ;
                    :25228          ;       OUTPUT
                    :25229          ;
                    :25230          ;-------------------------------------------------------------------------
                    :25231          ; R0    DELTA PC TEMP5(8)       R0(16)
                    :25232          ;       --------_-------- --------_--------
                    :25233          ; R1                    PC(32)
                    :25234          ;       --------_--------_--------_--------
                    :25235          ;-------------------------------------------------------------------------
                    :25236          ; (16) MEANS 16 BITS LONG
                    :25237          ;-------------------------------------------------------------------------
                    :25238          ;
                    :25239          ;*************************************************************************
                    :25240
                    :25241 CS.LOCC.SKPC.PACK.BE:
                    :25242          ;-------------------------------;
J 0BDE, 0986,5D12,0037,F847,00B9,6  :25243          M[TEMP5]_MB.AND.ZLIT16[OFF]     ;MTMP5_0'FILL'0'0
                    :25244
                    :25245 =
                    :25246 =0
                    :25247          ;0------------------------------;
                    :25248          PUSH,                           ;RET+1
                    :25249          R[R0]_M[TEMP5].OR.RB,           ;R0_0'FILL'R0(16)
U 0B96, 0884,5002,4034,0047,04BE,9  :25250          NEXT/CS.R1.GETS.PC              ;PUT PC INTO R1
                    :25251
                    :25252          ;1------------------------------;
                    :25253          R[TEMP1]_Q-M[PCBACK],           ;TEMP1_DELTA PC
U 0B97, 0485,900B,0670,4047,00A8,D  :25254          IR<2-0>?                        ;IF IR 1 NOT SET MATCHC
                    :25255
                    :25256 =101
                    :25257          ;101-----------------------------;
J 0A8D, 0480,0036,4030,0047,00FE,2  :25258          NEXT/IE.PACK.DONE               ;MATCHC DONE SO GO
                    :25259
                    :25260 =111
                    :25261          ;111-----------------------------;
                    :25262          R[R0]_RB.OR.(M[TEMP1].RL.24),   ;R0_DELTA PC'FILL'R0(16)
J 0A8F, 0084,13BE,4004,0047,00FE,2  :25263          NEXT/IE.PACK.DONE               ;GO TO CHARLIE
```

```
                                ;25264 .TOC     "       Character String                : LOCC and SKPC FPD Unpack routine"
                                ;25265
                                ;25266          ;*************************************************************************
                                ;25267          ; The input of the unpack code is the output of the pack code.
                                ;25268          ; The output of the unpack code is the input of the pack code.
                                ;25269          ; Refer to the pack routine banner a few pages back.
                                ;25270          ;*************************************************************************
                                ;25271
                                ;25272 .REGION/IRD1.R1L,IRD1.R1H
                                ;25273 =000
                                ;25274 CS.LOCC.SKPC.UNPACK:
                                ;25275          ;000-------------------------------;
U 0340, 0C86,52B7,0014,0047,00B3,0 ;25276       M[TEMP5]_R[R0].RR.16            ;TEMP5_GARBAGE'FILL
                                ;25277 =
                                ;25278
                                ;25279 .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                                ;25280 =00
                                ;25281          ;00-------------------------------;
                                ;25282          PUSH,                           ;RET+1
                                ;25283          M[TEMP5]_MB.AND.ZLIT0[0FF],     ;TEMP5_0'0'0'FILL
U 0B30, 0D86,5C12,0037,F847,04A9,A ;25284       NEXT/CS.EXP.FILLER              ;TEMP5_FILL'FILL'FILL'FILL
                                ;25285
                                ;25286          ;01-------------------------------;
                                ;25287          PUSH,                           ;RET+1
                                ;25288          M[TEMP1]_R[R0].RR.24,           ;TEMP1_GARBAGE'DELTA PC
U 0B31, 0086,12B7,0024,0047,04BF,8 ;25289       NEXT/CS.PC.GETS.R1              ;PC GETS R1
                                ;25290
                                ;25291          ;10-------------------------------;
U 0B32, 0580,1C12,1037,F847,00BD,F ;25292       Q_M[TEMP1].AND.ZLIT0[0FF]       ;QREG_original PC
                                ;25293 =
                                ;25294
                                ;25295          ;-------------------------------;
U 0BDF, 0081,9009,1030,0047,00BE,0 ;25296       Q_Q+M[PCBACK]
                                ;25297
                                ;25298          ;-------------------------------;
U 0BE0, 0986,CC37,0030,3047,00BE,1 ;25299       M[FPDOFFSET]_ZLIT0[6]           ;Load FPDOFFSET
                                ;25300
                                ;25301          ;-------------------------------;
                                ;25302          M[TEMP0]_R[R0],                 ;MTMP0 GETS R0 SO CAN ZERO EXTEND
U 0BE1, 0486,05EE,4674,0047,00AA,9 ;25303       IR<2-0>?                        ;IF MATCHC BIT 1 IS NOT SET
                                ;25304
                                ;25305 =001
                                ;25306          ;001------------------------------;
                                ;25307          M[TEMP1]_R[R2],                 ;TEMP1_R2
U 0AA9, 0886,15BE,4034,8047,00BE,2 ;25308       NEXT/CS.MAT.UNP.L.S             ;MATCHC SO RESTORE TEMP1
                                ;25309
                                ;25310 =011
                                ;25311          ;011------------------------------;
                                ;25312          R[R0]_D_ZEXT(M[TEMP0]),         ;GPR0 & DREG get length,
                                ;25313          SIZE[WORD],IR<2-0>?,            ;which inst are we?
U 0AAB, 0C84,059E,6654,0047,00A4,6 ;25314       NEXT/CS.OS.LOC
                                ;25315 =
```

```
                                        ;25316 CS.MAT.UNP.L.S:
                                        ;25317          ;--------------------------------;
                                        ;25318          M[TEMP1] ZEXT(MB),              ;(INCASE MATCHC
0) 0BE2, 0086,159E,4010,0047,00BE,3     ;25319          SIZE[WORD]                      ;ZERO EXTEND BY A WORD
                                        ;25320
                                        ;25321          ;--------------------------------;
0) 0BE3, 0880,05BE,4034,C4A7,00BE,4     ;25322          VA_R[R3]                        ;VA GETS RESTORED
                                        ;25323
                                        ;25324          ;--------------------------------;
                                        ;25325          R[R0]_D ZEXT(M[TEMP0]),         ;R0,D GETS LENGTH
0) 0BE4, 0884,059E,6014,0047,00A4,6     ;25326          SIZE[WORD],NEXT/CS.OS.LOC
```

```
                        ;25327 .TOC     "       Character String                : MATCHC FPD Unpack routine"
                        ;25328
                        ;25329          ;*************************************************************************
                        ;25330          ;       INPUT
                        ;25331          ;
                        ;25332          ;                       R0              R0(32)
                        ;25333          ;                       ------- ------- ------- -------
                        ;25334          ;                       R1              PC(32)(ALREADY)
                        ;25335          ;                       ------- ------- ------- -------
                        ;25336          ;                       R2      /\PC(8) ?(7)C(1)        R2(16)
                        ;25337          ;                       ------- ------- ------- -------
                        ;25338          ;                       R3              VA(32)(ALREADY)
                        ;25339          ;                       ------- ------- ------- -------
                        ;25340          ;
                        ;25341          ; Note:                  /\ := delta     (x) := x bits in length
                        ;25342          ;                       C := CODE       ? := garbage
                        ;25343          ;
                        ;25344          ;       OUTPUT          R0              LENGTH
                        ;25345          ;                       R1              =PC
                        ;25346          ;                       MTMP10          =R0
                        ;25347          ;                       PC              =R1
                        ;25348          ;                       VA              =R3
                        ;25349          ;                       DELTA PC        IN R2
                        ;25350          ;                       CODE            16TH BIT OF R2
                        ;25351          ;*************************************************************************
                        ;25352
                        ;25353 .REGION/IRD1.R1L,IRD1.R1H
                        ;25354 =000
                        ;25355 CS.MATCHC.UNPACK:
                        ;25356          ;000--------------------------------;
0348, 0480,05BE,46F4,8047,00B3,4        ;25357          WB_R[R2],WB<31-30>?             ;Are we MATCHC, LOCC, or CMPC3?
                        ;25358 =
                        ;25359
                        ;25360 .REGION/CHAR.R1L,CHAR.R1H/CHAR.R2L,CHAR.R2H/CHAR.R3L,CHAR.R3H
                        ;25361 =00
                        ;25362          ;00--------------------------------;
                        ;25363          VA_R[R3],                       ;VA GETS R3
0B34, 0080,05BE,4034,C4A7,00BE,5        ;25364          NEXT/CS.MAT.UNP.1               ;MATCHC
                        ;25365
                        ;25366          ;01--------------------------------;
                        ;25367          M[TEMP8]_0LIT24[13F],           ;TEMP8_3FFFFFFF
0B35, 0986,8EB7,0039,F847,0034,0        ;25368          NEXT/CS.LOCC.SKPC.UNPACK        ;LOCC
                        ;25369
                        ;25370          ;10--------------------------------;
                        ;25371          M[TEMP6]_0_FLAGS_R[R2],         ;TEMP6,0,FLAGS, GETS R2
0B36, 0086,65BE,5034,8307,00BD,1        ;25372          NEXT/CS.CMP.UNPACK.BEGIN        ;CMPC3
                        ;25373 =
```

```
                                        ;25374 CS.MAT.UNP.1:
                                        ;25375          ;--------------------------------;
U 0BE5, 0C80,05BE,4034,4487,00BE,6      ;25376          PC_R[R1]                        ;PC GETS R1
                                        ;25377
                                        ;25378          ;--------------------------------;
U 0BE6, 0886,15BE,4034,8047,00BE,7      ;25379          M[TEMP1]_R[R2]                  ;TEMP1_R2
                                        ;25380
                                        ;25381          ;--------------------------------;
                                        ;25382          M[TEMP1] ZEXT(MB),SIZE[WORD],   ;TEMP1 is restored, reenter LOCC
U 0BE7, 0086,159E,4010,0047,008A,1      ;25383          NEXT/CS.MAT.SET.FPDOFF.-1       ;preliminary flow
```

B 1	MICRO2 1M(01)	J 5	MICRO2 1M(01)
C 1	MICRO2 1M(01)	K 5	MICRO2 1M(01)
D 1	MICRO2 1M(01)	L 5	MICRO2 1M(01)
E 1	MICRO2 1M(01)	M 5	MICRO2 1M(01)
F 1	MICRO2 1M(01)	N 5	MICRO2 1M(01)
G 1	MICRO2 1M(01)	B 6	MICRO2 1M(01)
H 1	MICRO2 1M(01)	C 6	MICRO2 1M(01)
I 1	MICRO2 1M(01)	D 6	MICRO2 1M(01)
J 1	MICRO2 1M(01)	E 6	MICRO2 1M(01)
K 1	MICRO2 1M(01)	F 6	MICRO2 1M(01)
L 1	MICRO2 1M(01)	G 6	MICRO2 1M(01)
M 1	MICRO2 1M(01)	H 6	MICRO2 1M(01)
N 1	MICRO2 1M(01)	I 6	MICRO2 1M(01)
B 2	MICRO2 1M(01)	J 6	MICRO2 1M(01)
C 2	MICRO2 1M(01)	K 6	MICRO2 1M(01)
D 2	MICRO2 1M(01)	L 6	MICRO2 1M(01)
E 2	MICRO2 1M(01)	M 6	MICRO2 1M(01)
F 2	MICRO2 1M(01)	N 6	MICRO2 1M(01)
G 2	MICRO2 1M(01)	B 7	MICRO2 1M(01)
H 2	MICRO2 1M(01)	C 7	MICRO2 1M(01)
I 2	MICRO2 1M(01)	D 7	MICRO2 1M(01)
J 2	MICRO2 1M(01)	E 7	MICRO2 1M(01)
K 2	MICRO2 1M(01)	F 7	MICRO2 1M(01)
L 2	MICRO2 1M(01)	G 7	MICRO2 1M(01)
M 2	MICRO2 1M(01)	H 7	MICRO2 1M(01)
N 2	MICRO2 1M(01)	I 7	MICRO2 1M(01)
B 3	MICRO2 1M(01)	J 7	MICRO2 1M(01)
C 3	MICRO2 1M(01)	K 7	MICRO2 1M(01)
D 3	MICRO2 1M(01)	L 7	MICRO2 1M(01)
E 3	MICRO2 1M(01)	M 7	MICRO2 1M(01)
F 3	MICRO2 1M(01)	N 7	MICRO2 1M(01)
G 3	MICRO2 1M(01)	B 8	MICRO2 1M(01)
H 3	MICRO2 1M(01)	C 8	MICRO2 1M(01)
I 3	MICRO2 1M(01)	D 8	MICRO2 1M(01)
J 3	MICRO2 1M(01)	E 8	MICRO2 1M(01)
K 3	MICRO2 1M(01)	F 8	MICRO2 1M(01)
L 3	MICRO2 1M(01)	G 8	MICRO2 1M(01)
M 3	MICRO2 1M(01)	H 8	MICRO2 1M(01)
N 3	MICRO2 1M(01)	I 8	MICRO2 1M(01)
B 4	MICRO2 1M(01)	J 8	MICRO2 1M(01)
C 4	MICRO2 1M(01)	K 8	MICRO2 1M(01)
D 4	MICRO2 1M(01)	L 8	MICRO2 1M(01)
E 4	MICRO2 1M(01)	M 8	MICRO2 1M(01)
F 4	MICRO2 1M(01)	N 8	MICRO2 1M(01)
G 4	MICRO2 1M(01)	B 9	MICRO2 1M(01)
H 4	MICRO2 1M(01)	C 9	MICRO2 1M(01)
I 4	MICRO2 1M(01)	D 9	MICRO2 1M(01)
J 4	MICRO2 1M(01)	E 9	MICRO2 1M(01)
K 4	MICRO2 1M(01)	F 9	MICRO2 1M(01)
L 4	MICRO2 1M(01)	G 9	MICRO2 1M(01)
M 4	MICRO2 1M(01)	H 9	MICRO2 1M(01)
N 4	MICRO2 1M(01)	I 9	MICRO2 1M(01)
B 5	MICRO2 1M(01)	J 9	MICRO2 1M(01)
C 5	MICRO2 1M(01)	K 9	MICRO2 1M(01)
D 5	MICRO2 1M(01)	L 9	MICRO2 1M(01)
E 5	MICRO2 1M(01)	M 9	MICRO2 1M(01)
F 5	MICRO2 1M(01)	N 9	MICRO2 1M(01)
G 5	MICRO2 1M(01)	B 10	MICRO2 1M(01)
H 5	MICRO2 1M(01)	C 10	MICRO2 1M(01)
I 5	MICRO2 1M(01)	D 10	MICRO2 1M(01)

E 10	MICRO2 1M(01)		
F 10	MICRO2 1M(01)		
G 10	MICRO2 1M(01)		
H 10	MICRO2 1M(01)		
I 10	MICRO2 1M(01)		
J 10	MICRO2 1M(01)		
K 10	MICRO2 1M(01)		
L 10	MICRO2 1M(01)		
M 10	MICRO2 1M(01)		
N 10	MICRO2 1M(01)		
B 11	MICRO2 1M(01)		
C 11	MICRO2 1M(01)		
D 11	MICRO2 1M(01)		
E 11	MICRO2 1M(01)		
F 11	MICRO2 1M(01)		
G 11	MICRO2 1M(01)		
H 11	MICRO2 1M(01)		
I 11	MICRO2 1M(01)		
J 11	MICRO2 1M(01)		
K 11	MICRO2 1M(01)		
L 11	MICRO2 1M(01)		
M 11	MICRO2 1M(01)		
N 11	MICRO2 1M(01)		
B 12	MICRO2 1M(01)		
C 12	MICRO2 1M(01)		
D 12	MICRO2 1M(01)		
E 12	MICRO2 1M(01)		
F 12	MICRO2 1M(01)		
G 12	MICRO2 1M(01)		
H 12	MICRO2 1M(01)		
I 12	MICRO2 1M(01)		
J 12	MICRO2 1M(01)		
K 12	MICRO2 1M(01)		
L 12	MICRO2 1M(01)		
M 12	MICRO2 1M(01)		
N 12	MICRO2 1M(01)		
B 13	MICRO2 1M(01)		
C 13	MICRO2 1M(01)		
D 13	MICRO2 1M(01)		
E 13	MICRO2 1M(01)		
F 13	MICRO2 1M(01)		
G 13	MICRO2 1M(01)		
H 13	MICRO2 1M(01)		
I 13	MICRO2 1M(01)		
J 13	MICRO2 1M(01)		
K 13	MICRO2 1M(01)		
L 13	MICRO2 1M(01)		
M 13	MICRO2 1M(01)		
N 13	MICRO2 1M(01)		
B 14	MICRO2 1M(01)		
C 14	MICRO2 1M(01)		
D 14	MICRO2 1M(01)		
E 14	MICRO2 1M(01)		
F 14	MICRO2 1M(01)		
G 14	MICRO2 1M(01)		
H 14	MICRO2 1M(01)		
I 14	MICRO2 1M(01)		
J 14	MICRO2 1M(01)		
K 14	MICRO2 1M(01)		
L 14	MICRO2 1M(01)		

```
                                ;25384 .TOC   "       Character String                : FPD Subroutines"
                                ;25385
                                ;25386 CS.PC.GETS.R1:
                                ;25387          ;--------------------------------;
U 0BE8, 0080,05BE,40B4,4487,0000,1  ;25388          PC_R[R1],RETURN [1]             ;PC_GPR1, RETURN +1
                                ;25389
                                ;25390
                                ;25391
                                ;25392 CS.R1.GETS.PC:
                                ;25393          ;--------------------------------;
                                ;25394          R[R1]_M[PC],                    ;R1 GETS PC
U 0BE9, 0C85,A592,40B4,4047,0000,1  ;25395          RETURN [1]                      ;RET+1
                                ;25396
                                ;25397
                                ;25398
                                ;25399 CS.UNP.TMP10.IS.NEG:
                                ;25400          ;--------------------------------;
                                ;25401          R[TEMP2].SIZ_M[TEMP10],         ;TEMP2_FFFF'MTMP10(16)
U 0BEA, 0C82,A592,4010,8047,00BE,B  ;25402          SIZE[WORD]                      ;
                                ;25403          ;--------------------------------;
                                ;25404          M[TEMP10]_R[TEMP2],             ;MTMP10_FFFF'MTMP10
U 0BEB, 0C86,A5BE,40B0,8047,0000,1  ;25406          RETURN [1]                      ;RET+1
```

```
                                        ;25407 .TOC  ""  Character String  : Ird Rom Definition"
                                        ;25408
;25409  .NOBIN
;25410
;25411  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;25412       029: FPD [NOP][CS.CMP.UNPACK      ]                          [NOP][CS.CMP.UNPACK      ],                    ;CMPC3
;25413            IRD1[LOD][OS.RED             ]                          [LOD][OS.RED             ]
;25414  .OCODE
;25415       029: CNT0[NOP][CS.OS.MOVC3.CMPC3   ][CS.OS.MOVC3.CMPC3   ]  [NOP][CS.OS.MOVC3.CMPC3   ][CS.OS.MOVC3.CMPC3   ],
;25416            CNT1[NOP][CS.OS.MOVC3.CMPC3.1 ][CS.OS.MOVC3.CMPC3.1 ]  [NOP][CS.OS.MOVC3.CMPC3.1 ][CS.OS.MOVC3.CMPC3.1 ]
;25417
;25418  .ICODE
;25419       02D: FPD [NOP][CS.CMP.UNPACK      ]                          [NOP][CS.CMP.UNPACK      ],                    ;CMPC5
;25420            IRD1[LOD][OS.RED             ]                          [LOD][OS.RED             ]
;25421  .OCODE
;25422       02D: CNT0[NOP][CS.OS.MOVC5.CMPC5   ][CS.OS.MOVC5.CMPC5   ]  [NOP][CS.OS.MOVC5.CMPC5   ][CS.OS.MOVC5.CMPC5   ],
;25423            CNT1[NOP][CS.OS.MOVC5.CMPC5.1 ][CS.OS.MOVC5.CMPC5.1 ]  [NOP][CS.OS.MOVC5.CMPC5.1 ][CS.OS.MOVC5.CMPC5.1 ]
;25424
;25425  .ICODE
;25426       03A: FPD [NOP][CS.LOCC.SKPC.UNPACK ]                          [NOP][CS.LOCC.SKPC.UNPACK ],                   ;LOCC
;25427            IRD1[LOD][OS.RED             ]                          [LOD][OS.RED             ]
;25428  .OCODE
;25429       03A: CNT0[NOP][CS.OS.LOCC.SKPC     ][CS.OS.LOCC.SKPC     ]  [NOP][CS.OS.LOCC.SKPC     ][CS.OS.LOCC.SKPC     ],
;25430            CNT1[NOP][CS.OS.LOCC.SKPC.1   ][CS.OS.LOCC.SKPC.1   ]  [NOP][CS.OS.LOCC.SKPC.1   ][CS.OS.LOCC.SKPC.1   ]
;25431
;25432  .ICODE
;25433       039: FPD [NOP][CS.MATCHC.UNPACK    ]                          [NOP][CS.MATCHC.UNPACK    ],                   ;MATCHC
;25434            IRD1[LOD][OS.RED             ]                          [LOD][OS.RED             ]
;25435  .OCODE
;25436       039: CNT0[NOP][CS.OS.MATCHC        ][CS.OS.MATCHC        ]  [NOP][CS.OS.MATCHC        ][CS.OS.MATCHC        ],
;25437            CNT1[NOP][CS.OS.MATCHC.1      ][CS.OS.MATCHC.1      ]  [NOP][CS.OS.MATCHC.1      ][CS.OS.MATCHC.1      ]
;25438
;25439  .ICODE
;25440       028: FPD [NOP][CS.MOV.UNPACK       ]                          [NOP][CS.MOV.UNPACK       ],                   ;MOVC3
;25441            IRD1[LOD][OS.RED             ]                          [LOD][OS.RED             ]
;25442  .OCODE
;25443       028: CNT0[NOP][CS.OS.MOVC3.CMPC3   ][CS.OS.MOVC3.CMPC3   ]  [NOP][CS.OS.MOVC3.CMPC3   ][CS.OS.MOVC3.CMPC3   ],
;25444            CNT1[NOP][CS.OS.MOVC3.CMPC3.1 ][CS.OS.MOVC3.CMPC3.1 ]  [NOP][CS.OS.MOVC3.CMPC3.1 ][CS.OS.MOVC3.CMPC3.1 ]
;25445
;25446  .ICODE
;25447       02C: FPD [NOP][CS.MOV.UNPACK       ]                          [NOP][CS.MOV.UNPACK       ],                   ;MOVC5
;25448            IRD1[LOD][OS.RED             ]                          [LOD][OS.RED             ]
;25449  .OCODE
;25450       02C: CNT0[NOP][CS.OS.MOVC5.CMPC5   ][CS.OS.MOVC5.CMPC5   ]  [NOP][CS.OS.MOVC5.CMPC5   ][CS.OS.MOVC5.CMPC5   ],
;25451            CNT1[NOP][CS.OS.MOVC5.CMPC5.1 ][CS.OS.MOVC5.CMPC5.1 ]  [NOP][CS.OS.MOVC5.CMPC5.1 ][CS.OS.MOVC5.CMPC5.1 ]
;25452
;25453  .ICODE
;25454       02E: FPD [NOP][CS.MOV.UNPACK       ]                          [NOP][CS.MOV.UNPACK       ],                   ;MOVTC
;25455            IRD1[LOD][OS.RED             ]                          [LOD][OS.RED             ]
;25456  .OCODE
;25457       02E: CNT0[NOP][CS.OS.MOVTC.MOVTUC  ][CS.OS.MOVTC.MOVTUC  ]  [NOP][CS.OS.MOVTC.MOVTUC  ][CS.OS.MOVTC.MOVTUC  ],
;25458            CNT1[NOP][CS.OS.MOVTC.MOVTUC.1][CS.OS.MOVTC.MOVTUC.1]  [NOP][CS.OS.MOVTC.MOVTUC.1][CS.OS.MOVTC.MOVTUC.1]
```

```
;25459  .ICODE  ;       OPS             REG                     MEM             OPS         FPA REG                 FPA MEM
;25460        02F: FPD [NOP][CS.MOV.UNPACK      ]                             [NOP][CS.MOV.UNPACK      ],                    ;MOVTUC
;25461             IRD1[LOD][OS.RED             ]                             [LOD][OS.RED             ]
;25462  .OCODE
;25463        02F: CNT0[NOP][CS.OS.MOVTC.MOVTUC  ][CS.OS.MOVTC.MOVTUC  ]  [NOP][CS.OS.MOVTC.MOVTUC  ][CS.OS.MOVTC.MOVTUC  ],
;25464             CNT1[NOP][CS.OS.MOVTC.MOVTUC.1][CS.OS.MOVTC.MOVTUC.1]  [NOP][CS.OS.MOVTC.MOVTUC.1][CS.OS.MOVTC.MOVTUC.1]
;25465
;25466  .ICODE
;25467        02A: FPD [NOP][CS.SC.SPAN.UNPACK   ]                             [NOP][CS.SC.SPAN.UNPACK   ],                    ;SCANC
;25468             IRD1[LOD][OS.RED              ]                             [LOD][OS.RED              ]
;25469  .OCODE
;25470        02A: CNT0[NOP][CS.OS.SCANC.SPANC   ][CS.OS.SCANC.SPANC   ]  [NOP][CS.OS.SCANC.SPANC   ][CS.OS.SCANC.SPANC   ],
;25471             CNT1[NOP][CS.OS.SCANC.SPANC.1 ][CS.OS.SCANC.SPANC.1 ]  [NOP][CS.OS.SCANC.SPANC.1 ][CS.OS.SCANC.SPANC.1 ]
;25472
;25473  .ICODE
;25474        03B: FPD [NOP][CS.LOCC.SKPC.UNPACK ]                             [NOP][CS.LOCC.SKPC.UNPACK ],                    ;SKPC
;25475             IRD1[LOD][OS.RED              ]                             [LOD][OS.RED              ]
;25476  .OCODE
;25477        03B: CNT0[NOP][CS.OS.LOCC.SKPC     ][CS.OS.LOCC.SKPC     ]  [NOP][CS.OS.LOCC.SKPC     ][CS.OS.LOCC.SKPC     ],
;25478             CNT1[NOP][CS.OS.LOCC.SKPC.1   ][CS.OS.LOCC.SKPC.1   ]  [NOP][CS.OS.LOCC.SKPC.1   ][CS.OS.LOCC.SKPC.1   ]
;25479
;25480  .ICODE
;25481        02B: FPD [NOP][CS.SC.SPAN.UNPACK   ]                             [NOP][CS.SC.SPAN.UNPACK   ],                    ;SPANC
;25482             IRD1[LOD][OS.RED              ]                             [LOD][OS.RED              ]
;25483  .OCODE
;25484        02B: CNT0[NOP][CS.OS.SCANC.SPANC   ][CS.OS.SCANC.SPANC   ]  [NOP][CS.OS.SCANC.SPANC   ][CS.OS.SCANC.SPANC   ],
;25485             CNT1[NOP][CS.OS.SCANC.SPANC.1 ][CS.OS.SCANC.SPANC.1 ]  [NOP][CS.OS.SCANC.SPANC.1 ][CS.OS.SCANC.SPANC.1 ]
```

```
;25486  .TOC   "      Character String                : Dsize Rom Definition"
;25487  .DCODE
;25488
;25489  029:    SIZE [WORD] [BYTE] [BYTE] [   0] [   0] [   0]  ;CMPC3
;25490
;25491  02D:    SIZE [WORD] [BYTE] [BYTE] [WORD] [BYTE] [   0]  ;CMPC5
;25492
;25493  03A:    SIZE [BYTE] [WORD] [BYTE] [   0] [   0] [   0]  ;LOCC
;25494
;25495  039:    SIZE [WORD] [BYTE] [WORD] [BYTE] [   0] [   0]  ;MATCHC
;25496
;25497  028:    SIZE [WORD] [BYTE] [BYTE] [   0] [   0] [   0]  ;MOVC3
;25498
;25499  02C:    SIZE [WORD] [BYTE] [BYTE] [WORD] [BYTE] [   0]  ;MOVC5
;25500
;25501  02E:    SIZE [WORD] [BYTE] [BYTE] [BYTE] [WORD] [BYTE]  ;MOVTC
;25502
;25503  02F:    SIZE [WORD] [BYTE] [BYTE] [BYTE] [WORD] [BYTE]  ;MOVTUC
;25504
;25505  02A:    SIZE [WORD] [BYTE] [BYTE] [BYTE] [   0] [   0]  ;SCANC
;25506
;25507  03B:    SIZE [BYTE] [WORD] [BYTE] [   0] [   0] [   0]  ;SKPC
;25508
;25509  02B:    SIZE [WORD] [BYTE] [BYTE] [BYTE] [   0] [   0]  ;SPANC
;25510
;25511  .UCODE
                                        ;25512 .BIN
```

```
                                        ;25513 .TOC "DECMAL.MIC"
                                        ;25514 .TOC "REVISION 29.0"
                                        ;25515 ;       Jeff Peng, Jim Heom, Gerard Koeckhoven, John DeRosa, CHARLIE MCDOWELL
                                        ;25516
;25517  .NOBin
;25518
;25519  .TOC "  Revision History"
;25520
;25521  ; Rev.  Explanation
;25522
;25523  ; 29    The location of u-code 1312 & 1313 are not used.
;25524  ; 28    Set mm.noint for addp4 -(works for sub also) when writing
;25525  ;       across page boundary in reverse direction
;25526  ;       Merged some locations into common code to save 4 locations
;25527  ; 27    Set MM.NOINT during the write of CVTPL to avoid problem with
;25528  ;       the Write Crossing Page Boundry routine.
;25529  ; 26    Force some addresses and make some documentation changes
;25530  ; 25    Fix ASHP to take a reserved operand abort when dest len > 31.
;25531  ;       Fix CVTPS instruction to clear the Z-bit when SRC len > Dest len
;25532  ; 24    Whenever the src len > dest len and the source string is not equal
;25533  ;       to zero and the first byte after a possible zero byte is not an illegal
;25534  ;       ascii character then a reserved operand fault will occur.
;25535  ; 23    Fix CVTTP and CVTSP  illegal byte detection fixed in 22
;25536  ;       Fix CVTPS instruction to clear the Z-bit when source lenght = 1
;25537  ; 22    29-MAY-1980
;25538  ;       Fix CVTTP to zero upper nibble of 0 length result if a page fault
;25539  ;       occurs trying to write the result.
;25540  ;       12-JUNE-1980
;25541  ;       Fix CVTLP to always properly set/clear Z if overflow occurs.
;25542  ;       Fix CVTPS and CVTPT to properly set CC's if destination length = 0.
;25543  ;       18-JUNE-1980
;25544  ;       Fix CVTTP and CVTSP to take a reserved operand fault for all illegal
;25545  ;       bytes that overflow.
;25546  ;       Fix CVTPS and CVTPT to set Z if the srclen > dstlen and dstlen is even.
;25547  ; 21    Fix converts to save flags if interrupted with FPD set.
;25548  ;       Fix DIVP to take reserved operand if divide loop hangs do to
;25549  ;       illegal BCD digits.
;25550  ; 20    Initial release.
                                        ;25551          .BIN
```

```
;25552 .TOC "  Decimal String                    : MOVP"
;25553
;25554          ;*****************************************************************************
;25555          ;
;25556          ;       MOVP    len.rw, srcaddr.ab, dstaddr.ab
;25557          ;
;25558          ; The instruction initializatizes, READs data, masks in the sign,
;25559          ; WRITEs data, checks for zero destination, etc., then sets CC.
;25560          ; CMPP3 does operand fetch and preliminary initialization in MOVP code.
;25561          ;
;25562          ;       Input           MDR             Length
;25563          ;
;25564          ;       Resources       GPR1            Base address, source
;25565          ;                       GPR3            Base address, destination
;25566          ;                       TEMP1           Misc. storage
;25567          ;                       TEMP3-TEMP6     Packed Decimal data, memory
;25568          ;                                       format, LSB in byte#3 of TEMP3,
;25569          ;                                       MSB in byte#0 of TEMP6
;25570          ;                       FPDOFFSET       ZLITO[17.]
;25571          ;                       MTEMP8          Source & Destination Length
;25572          ;                       STEPC           I/O counter
;25573          ;                       MDR             READs
;25574          ;                       WDR             WRITEs
;25575          ;                       VA              I/O address
;25576          ;                       DREG            Misc. temp. storage
;25577          ;                       QREG            Ditto
;25578          ;                       PSL<C>          _1 so DS.WRITE thinks dest. is
;25579          ;                                       odd.  Legal even sources will
;25580          ;                                       be written properly, and illegal
;25581          ;                                       even sources (high nibble >0)
;25582          ;                                       will have the unpredictable
;25583          ;                                       result of writing the dest. with
;25584          ;                                       a nonzero high nibble.
;25585          ;                       FLAG0           Used in DS.WRITE, masks PSL<C>
;25586          ;                       FLAG1           Value of PSL<C> before IRD1
;25587          ;                       FLAG2           <Z>image, initially 1
;25588          ;                       FLAG3           <N>image, initially 0
;25589          ;
;25590          ;       Output          GPR0            0
;25591          ;                       GPR1            Base add., source 1
;25592          ;                       GPR2            0
;25593          ;                       GPR3            Base add., destination
;25594          ;                       CC              V_0, NZ modified
;25595          ;
;25596          ;*****************************************************************************
```

```
                                ;25597 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;25598 DS.MOVP:
                                ;25599          ;--------------------------------;
                                ;25600          NEXT/OS.ADD,SIZE[WORD],         ;TEMP1_Length,
                                ;25601          R[TEMP1]_SEXT(M[MDR]),          ;READ source address
U 03B7, 0485,2E5E,0190,4047,0012,0 ;25602       LOD INC BRA?
                                ;25603
                                ;25604 =0
                                ;25605 DS.MOVP.2:
                                ;25606          ;0-------------------------------;
                                ;25607          PUSH,NEXT/OS.ADD,LOD INC BRA?,  ;STEPC_D_L/2 for DS.READ, READ
U 0312, 0880,1001,A1BD,8107,0412,0 ;25608       STEPC_D_M[TEMP1].SR.1           ;dest. address
                                ;25609
                                ;25610          ;1-------------------------------;
                                ;25611          WB_M[TEMP1].ANDNOT.ZLITO[31.],  ;Is the length ok?
U 0313, 0D80,1C13,8A30,F847,0101,8 ;25612       WX.EQ.0?
                                ;25613
                                ;25614 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;25615 =0
                                ;25616          ;0-------------------------------;
U 1018, 0C80,0036,4030,0047,00FF,8 ;25617       NEXT/IE.OPER.FAULT              ;**Reserved operand fault**
                                ;25618
                                ;25619          ;1-------------------------------;
U 1019, 00EC,002C,7974,4047,0104,4 ;25620       R[R1]_Q Q_D,SET FPD,PSL<C>?     ;GPR1_src. base address, what's current
                                ;25621                                          ;PSL<C>? (Q_garbage)
                                ;25622
                                ;25623 =0
                                ;25624          ;0-------------------------------;
                                ;25625          CLEAR FLAG1,R[R3]_M[MDR],       ;FLAG1_0, R3_dest. address, set PSL<C>
U 1044, 080D,2592,4034,C047,0100,6 ;25626       NEXT/DS.MOVP.2.PSLC
                                ;25627
                                ;25628          ;1-------------------------------;
                                ;25629          SET FLAG1,R[R3]_M[MDR],         ;FLAG1_1, PSL<C> is set, which inst.
U 1045, 0C4D,2592,4674,C047,0100,4 ;25630       IR<2-0>?,NEXT/DS.MOVP.INIT      ;are we?
                                ;25631
                                ;25632 DS.MOVP.2.PSLC:
                                ;25633          ;--------------------------------;
U 1006, 0580,0C37,0670,08A7,0100,4 ;25634       CC_ZLITO[1],IR<2-0>?            ;PSL<C>_1, which inst. are we?
                                ;25635
                                ;25636 =1*0
                                ;25637 DS.MOVP.INIT:
                                ;25638          ;1*0-----------------------------;
                                ;25639          M[TEMP8]_R[TEMP1],              ;MOVP[34], MTEMP8_dest. length
U 1004, 0486,85BE,4030,4047,0100,D ;25640       NEXT/DS.MOVP.INIT.2
                                ;25641
                                ;25642          ;1*1-----------------------------;
                                ;25643          M[TEMP8]_R[TEMP1],              ;CMPP3[35], load source length
U 1005, 0486,85BE,4030,4047,0100,E ;25644       NEXT/DS.CMPP3
```

```
                                ;25645 DS.MOVP.INIT.2:
                                ;25646 DS.MOVP.REE:
                                ;25647          ;-------------------------------;
                                ;25648          M[FPDOFFSET]_ZLIT0[17.],        ;Load FPDOFFSET, anything
                                ;25649          COUNT OR INT_TIMER?,SIZE[LONG], ;pending?
U 100D, 0586,CC37,0B20,8847,090i,C 350*         ;25650          NEXT/DS.MOVP.STATE.INT
                                ;25651
                                ;25652 =000
                                ;25653 =011
                                ;25654          ;011-----------------------------;
                                ;25655          STEPC_Q_M[TEMP8].SR.1,ALUS?,    ;RETURN-1 from DS.READ, STEPC_Q_L/2,
U 101B, 0480,8E52,9B70,1907,0100,1              ;25656          NEXT/DS.MOVP.SIGN               ;what is sign?
                                ;25657
                                ;25658 DS.MOVP.STATE.INT:
                                ;25659          ;100-----------------------------;
                                ;25660          PUSH,VA_D+R[R1]+1,              ;Nothing pending or timer service,
U 101C, 0C80,00A1,0034,44A7,0505,D              ;25661          NEXT/DS.READ                    ;VA points past sign byte, read src.
                                ;25662
                                ;25663          ;101---- ----------------------;
                                ;25664          PUSH,INTPEND OR TIMER?,         ;Something pending, return -1 or
U 101D, 0480,0036,4AF0,0047,04F7,0              ;25665          NEXT/IE.SERV.IP.TS2             ;branch to IE.PACK.DONE
                                ;25666 =
                                ;25667
                                ;25668
                                ;25669          ;*********************************************************************
                                ;25670          ; Here after reading source.  =01 is positive source, =11 is
                                ;25671          ; negative source.
                                ;25672          ;*********************************************************************
                                ;25673 =01
                                ;25674 DS.MOVP.SIGN:
                                ;25675          ;01-----------------------------;
                                ;25676          M[TEMP3]_MB.OR.ZLIT24[0C],      ;Mask positive sign, <N>image_0
U 1001, 0D1E,3C92,4030,6047,0105,1              ;25677          CLEAR FLAG3,NEXT/DS.MOVP.WRITE
                                ;25678
                                ;25679          ;11-----------------------------;
                                ;25680          M[TEMP3]_MB.OR.ZLIT24[0D],      ;Ditto, but negative source
U 1003, 015E,3C92,4030,6847,0105,1              ;25681          SET FLAG3,NEXT/DS.MOVP.WRITE
                                ;25682
                                ;25683
                                ;25684 =0
                                ;25685          ;0-----------------------------;
                                ;25686          D_M[TEMP8].SR.1,FLAG2?,         ;Load DREG for potential sign byte
U 1050, 0080,8591,A4B0,0047,0100,8              ;25687          NEXT/DS.MOVP.FLAGS              ;rewrite, what is <Z>image?
                                ;25688
                                ;25689 DS.MOVP.WRITE:
                                ;25690          ;1-----------------------------;
                                ;25691          VA_R[R3],PUSH,SET FLAG2,        ;VA Base address, init. <N>image,
U 1051, 085C,058E,4034,C4A7,052C,E              ;25692          NEXT/DS.WRITE                   ;WRITE data
```

```
                                ;25693 .TOC "  Decimal String                  :       Setting CC"
                                ;25694
                                ;25695          ;*************************************************************************
                                ;25696          ; Dest. is written, FLAGS have CC data.
                                ;25697          ; 2 overlapping splits occur:
                                ;25698          ; 1) From above, FLAG2?: =0*0 is dest. is nonzero, <NZ> are ok.
                                ;25699          ;                        =1*0 is dest. is zero, check for -0 written
                                ;25700          ;                        to dest. and change FLAG3 & sign byte
                                ;25701          ;                        to positive.
                                ;25702          ; 2) From -0 change, FLAG3?: =0*0 is dest. is +0, <NZ> are ok.
                                ;25703          ;                            =0*1 is dest. is -0, dest._+0, FLAG3_0.
                                ;25704          ; We BUT to ASHP code after fixing sign nibble & FLAG3 if necessary.
                                ;25705          ;*************************************************************************
                                ;25706 =0*0
                                ;25707 DS.MOVP.FLAGS:
                                ;25708          ;0*0-------------------------;
                                ;25709          CLEAR FLAG1,FLAG1?,             ;BUT on old FLAG1 value (PSL<C>),
U 1008, 0C08,0036,45F0,0047,0100,0      ;25710          NEXT/DS.MOVP.EXIT               ;<V> mask_0
                                ;25711
                                ;25712          ;0*1-------------------------;
                                ;25713          WRITE ZLIT0[0C],SIZE[BYTE],     ;<N>image_0, dest._+0 in sign byte
U 1009, 0918,0C37,0000,65D8,0100,8      ;25714          CLEAR FLAG3,NEXT/DS.MOVP.FLAGS
                                ;25715
                                ;25716          ;1*0-------------------------;
                                ;25717          VA_D+R[R3],FLAG3?,              ;Load VA for possible sign byte READ,
U 100C, 0080,0021,04F4,C4A7,0100,8      ;25718          NEXT/DS.MOVP.FLAGS              ;Is dest. pos (=0*0) or neg (=0*1)?
                                ;25719 =
                                ;25720
                                ;25721
                                ;25722 =0*
                                ;25723 DS.MOVP.EXIT:
                                ;25724          ;0*-------------------------;
U 1000, 0884,0036,4034,8387,0138,E      ;25725          R[R2]_FLAGS,NEXT/DS.CCPCIRD1    ;GPR2_CCimage, finish in ASHP code
                                ;25726
                                ;25727          ;1*-------------------------;
U 1002, 0C00,0036,4030,0047,0100,0      ;25728          SET FLAG0,NEXT/DS.MOVP.EXIT
```

```
;25729 .TOC "  Decimal String                    : CMPP3"
;25730
;25731          ;*************************************************************************
;25732          ;
;25733          ; CMPP3 len.rw, src1addr.ab, src2addr.ab
;25734          ;
;25735          ; CMPP3 begins in MOVP code to evaluate operands and perform
;25736          ; preliminary initialization.
;25737          ;
;25738          ; CMPP4 initializes and enters CMPP3 code at DS.CMPP3: if the source
;25739          ; lengths are equal or the longer source has leading zeroes.  CMPP3
;25740          ; microcode will read the source base address from TEMP1 & TEMP2
;25741          ; instead of GPR1 & GPR3 because CMPP4 may have read leading bytes from
;25742          ; a source and bumped the "base address" up to avoid extra READs.
;25743          ;
;25744          ; Sources are read, signs remembered, and a !src.#2!-!src.#1!
;25745          ; subtraction done.
;25746          ;
;25747          ; In the Input banner, "*" means for CMPP4.
;25748          ;
;25749          ;       Input           GPR1            Source 1 address
;25750          ;                       GPR3            Source 2 address
;25751          ;                       FPD             =1
;25752          ;                       TEMP1*          Source 1 address (see above)
;25753          ;                       TEMP2*          Source 2 address (see above)
;25754          ;                       MTEMP8          Source Length
;25755          ;                       STEPC           L/2
;25756          ;                       DREG            L/2
;25757          ;                       Length          Checked for reserved operand
;25758          ;                                       fault
;25759          ;
;25760          ;       Resources       <Above, plus:>
;25761          ;                       FLAG0           Source 2 sign
;25762          ;                       FLAG1           Source 1 sign
;25763          ;                       FLAG2           =0 if result of source
;25764          ;                                       subtraction is zero (src=src),
;25765          ;                                       =1 if subtraction result is
;25766          ;                                       nonzero (src<>src)
;25767          ;                       FLAG3           =0 if Source 1 is zero
;25768          ;                                       =1 if Source 1 is nonzero
;25769          ;                       DREG            Misc. data
;25770          ;                       VA              READ addresses
;25771          ;                       MDR             Data
;25772          ;                       TEMP3-TEMP6     Holds source 2, memory format,
;25773          ;                                       sign nibble zero, LSB in byte#3
;25774          ;                                       of TEMP3, MSB in byte#0 of
;25775          ;                                       TEMP6
;25776          ;                       RTEMP10-RTEMP13 Source 1, memory format, sign
;25777          ;                                       nibble zero, LSB in byte#3 of
;25778          ;                                       TEMP3, MSB in byte#0 of TEMP6
;25779          ;                       FPDOFFSET       _ZLIT0[20.]
```

```
                    ;25780          ;       Output          GPR0            0
                    ;25781          ;                       GPR1            Base add., source 1
                    ;25782          ;                       GPR2            0
                    ;25783          ;                       GPR3            Base add., source 2
                    ;25784          ;                       CC              NZ modified, VC_0
                    ;25785          ;
                    ;25786          ;*******************************************************************************
```

```
                                        ;25787 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                        ;25788 DS.CMPP3:
                                        ;25789          ;--------------------------------;
                                        ;25790          M[FPDOFFSET]_ZLIT0[20.],         ;Load FPDOFFSET, anything pending?
                                        ;25791          COUNT OR INT TIMER?,SIZE[LONG], ;
U 100E, 0186,CC37,0B20,A047,0904,2 350* ;25792          NEXT/DS.CMPP.REE
                                        ;25793
                                        ;25794 =00
                                        ;25795          ;00--------------------------------;
                                        ;25796          CC_ZLIT0[0] ALUS?,SIZE[LONG],   ;RETURN-1 from Src.#2 READ, guess
U 1040, 0180,0C37,0B60,00A7,0106,1      ;25797          NEXT/DS.CMPP.SRC2.SIGN          ;src.#1>src.#2, what is src.#2 sign?
                                        ;25798
                                        ;25799          ;01--------------------------------;
                                        ;25800          R[TEMP10]_M[TEMP3],CLEAR FLAG3, ;RETURN-1 from Src.#1 READ, PUSH for
                                        ;25801          WX.NE.0?,NEXT/DS.CMPP.SRC1.1,   ;source 2 READ, move src.#1 into RTEMPs,
U 1041, 001C,3592,4A72,8047,0506,4      ;25802          PUSH                            ;check source#1 for zeroness
                                        ;25803
                                        ;25804 DS.CMPP.REE:
                                        ;25805          ;10--------------------------------;
                                        ;25806          VA_D+R[R1]+1,PUSH,NEXT/DS.READ, ;Nothing pending or timer, READ src.#1
U 1042, 0880,00A1,0674,44A7,0505,D      ;25807          IR<2-0>?                        ;spliting on opcode
                                        ;25808
                                        ;25809          ;11--------------------------------;
                                        ;25810          PUSH,INTPEND OR TIMER?,         ;Check for ints./timer, return-1 or
U 1043, 0480,0036,4AF0,0047,04F7,0      ;25811          NEXT/IE.SERV.IP.TS2             ;pack up
                                        ;25812
                                        ;25813
                                        ;25814 =0
                                        ;25815 DS.CMPP.SRC1.1:
                                        ;25816          ;0---------------------------------;
                                        ;25817          R[TEMP11]_M[TEMP4],WX.NE.0?,
U 1064, 0084,4592,4A72,C047,0107,C      ;25818          NEXT/DS.CMPP.SRC1.2
                                        ;25819
                                        ;25820          ;1---------------------------------;
                                        ;25821          R[TEMP11]_M[TEMP4],SET FLAG3,   ;Source 1 is nonzero, don't
U 1065, 0C5C,4592,4032,C047,0107,D      ;25822          NEXT/DS.CMPP.SRC1.2A            ;BUT on this longword
                                        ;25823
                                        ;25824 =0
                                        ;25825 DS.CMPP.SRC1.2:
                                        ;25826          ;0---------------------------------;
                                        ;25827          R[TEMP12]_M[TEMP5],WX.NE.0?,
U 107C, 0084,5592,4A73,0047,0108,C      ;25828          NEXT/DS.CMPP.SRC1.3
                                        ;25829
                                        ;25830 DS.CMPP.SRC1.2A:
                                        ;25831          ;1---------------------------------;
                                        ;25832          R[TEMP12]_M[TEMP5],SET FLAG3,   ;Source 1 is nonzero, don't BUT on this
U 107D, 0C5C,5592,4033,0047,0108,D      ;25833          NEXT/DS.CMPP.SRC1.3A            ;longword
```

```
                                        ;25834 =0
                                        ;25835 DS.CMPP.SRC1.3:
                                        ;25836          ;0------------------------------;
                                        ;25837          R[TEMP13]_M[TEMP6],WX.NE.0?,
U 108C, 0484,6592,4A73,4047,010A,4      ;25838          NEXT/DS.CMPP.SRC1.4
                                        ;25839
                                        ;25840 DS.CMPP.SRC1.3A:
                                        ;25841          ;1------------------------------;
                                        ;25842          R[TEMP13]_M[TEMP6],SET FLAG3,   ;Source 1 was nonzero, don't BUT on this
U 108D, 005C,6592,4B73,5847,0104,9      ;25843          ALUS?,NEXT/DS.CMPP.SRC1.SGN     ;portion, what is source 1 sign?
                                        ;25844
                                        ;25845 =0
                                        ;25846 DS.CMPP.SRC1.4:
                                        ;25847          ;0------------------------------;
U 10A4. 0480,0036,4B70,1847,0104,9      ;25848          ALUS?,NEXT/DS.CMPP.SRC1.SGN     ;What is source 1 sign?
                                        ;25849
                                        ;25850          ;1------------------------------;
U 10A5, 0458,0036,4B70,1847,0104,9      ;25851          SET FLAG3,ALUS?                 ;Source 1 is nonzero, ditto
                                        ;25852
                                        ;25853
                                        ;25854 =01
                                        ;25855 DS.CMPP.SRC1.SGN:
                                        ;25856          ;01-----------------------------;
                                        ;25857          CLEAR FLAG1,IR<2-0>?,           ;Positive, FLAG1_0, load STEPC & DREG
                                        ;25858          STEPC_D_M[TEMP8].SR.1,          ;for source 2 READ, which inst. are we?
U 1049, 0408,8001,A67D,8107,0101,5      ;25859          NEXT/DS.CMPP.SRC2
                                        ;25860
                                        ;25861          ;11-----------------------------;
                                        ;25862          SET FLAG1,IR<2-0>?,             ;Negative, FLAG1_1, ditto
U 104B, 0C48,8001,A67D,8107,0101,5      ;25863          STEPC_D_M[TEMP8].SR.1
                                        ;25864
                                        ;25865 =101
                                        ;25866 DS.CMPP.SRC2:
                                        ;25867          ;101----------------------------;
                                        ;25868          VA_D+R[R3]+1,CLEAR FLAG2,       ;CMPP3[35], Point VA past sign byte,
U 1015, 0010,00A1,0034,C4A7,0105,D      ;25869          NEXT/DS.READ                    ;sub. result FLAG_0 for when we return
                                        ;25870
                                        ;25871          ;111----------------------------;
                                        ;25872          VA_D+R[TEMP2]+1,CLEAR FLAG2,    ;CMPP4[37], Point VA past sign byte,
U 1017, 0010,00A1,0030,84A7,0105,D      ;25873          NEXT/DS.READ                    ;sub. result FLAG_0 for when we return
```

```
                                        ;25874          ;*************************************************************************
                                        ;25875          ; Sources have been read, here on source 2 sign split.
                                        ;25876          ; =01 is source 2 is positive.
                                        ;25877          ; =11 is source 2 is negative.
                                        ;25878          ; We subtract the LONGWORDs looking for a nonzero result.  Once we
                                        ;25879          ; have one, we don't split on the remaining subtracts.
                                        ;25880          ;*************************************************************************
                                        ;25881 =01
                                        ;25882 DS.CMPP.SRC2.SIGN:
                                        ;25883          ;01-----------------------------;
                                        ;25884          WB_(M[TEMP3]-R[TEMP10]).BCD,    ;Begin subtracting, check for
                                        ;25885          WX.NE.0?,CLEAR FLAG0,           ;nonzero result
U 1061, 0C00,3000,4A62,8047,090A,C 358* ;25886          NEXT/DS.CMPP.SUB.1
                                        ;25887
                                        ;25888          ;11-----------------------------;
                                        ;25889          SET FLAG0,WX.NE.0?,             ;Ditto
U 1063, 0440,3000,4A62,8047,090A,C 358* ;25890          WB_(M[TEMP3]-R[TEMP10]).BCD
                                        ;25891
                                        ;25892 =0
                                        ;25893 DS.CMPP.SUB.1:
                                        ;25894          ;0-----------------------------;
                                        ;25895          WB_(M[TEMP4]-R[TEMP11]-ALKC).BCD,       ;Last sub=0, next TEMP, BORROW
U 10AC, 0080,4040,4A62,C047,090B,4 358* ;25896          WX.NE.0?,NEXT/DS.CMPP.SUB.2
                                        ;25897
                                        ;25898          ;1-----------------------------;
                                        ;25899          SET FLAG2,                      ;Sub>0, next TEMP, BORROW, don't split
                                        ;25900          WB_(M[TEMP4]-R[TEMP11]-ALKC).BCD,
U 10AD, 0C50,4040,4022,C047,010B,5      ;25901          NEXT/DS.CMPP.SUB.2A
                                        ;25902
                                        ;25903 =0
                                        ;25904 DS.CMPP.SUB.2:
                                        ;25905          ;0-----------------------------;
                                        ;25906          WB_(M[TEMP5]-R[TEMP12]-ALKC).BCD,       ;Sub=0, next TEMP, carry BORROW
U 10B4, 0880,5040,4A63,0047,090B,C 358* ;25907          WX.NE.0?,NEXT/DS.CMPP.SUB.3
                                        ;25908
                                        ;25909 DS.CMPP.SUB.2A:
                                        ;25910          ;1-----------------------------;
                                        ;25911          SET FLAG2,                      ;Sub>0, carry BORROW, don't split
                                        ;25912          WB_(M[TEMP5]-R[TEMP12]-ALKC).BCD,
U 10B5, 0450,5040,4023,0047,010B,D      ;25913          NEXT/DS.CMPP.SUB.3A
                                        ;25914
                                        ;25915
                                        ;25916 =0
                                        ;25917 DS.CMPP.SUB.3:
                                        ;25918          ;0-----------------------------;
                                        ;25919          WB_(M[TEMP6]-R[TEMP13]-ALKC).BCD,       ;Sub=0, last TEMP, carry BORROW,
                                        ;25920          WX.NE.0?,                       ;save BORROW in ALUS
U 10BC, 0080,6040,4A63,50C7,090C,2 358* ;25921          ALUS_UNSGN,NEXT/DS.CMPP.SUB.LAS
                                        ;25922
                                        ;25923 DS.CMPP.SUB.3A:
                                        ;25924          ;1-----------------------------;
                                        ;25925          SET FLAG2,                      ;Sub>0, last TEMP, carry&ALUS_BORROW
                                        ;25926          WB_(M[TEMP6]-R[TEMP13]-ALKC).BCD,
U 10BD, 0450,6040,4023,50C7,010C,2      ;25927          ALUS_UNSGN
```

```
                                ;25928 =0
                                ;25929 DS.CMPP.SUB.LAS:
                                ;25930          ;0--------------------------------;
                                ;25931          R[R2]_0,ALUS?,                  ;Result=0, begin to finish inst.,
U 10C2, 0484,05B7,0B74,9847,0106,D ;25932       NEXT/DS.CMPP.FLAGS              ;Was there a BORROW?
                                ;25933
                                ;25934          ;1--------------------------------;
                                ;25935          SET FLAG2,R[R2]_0,              ;Result>0, ditto
U 10C3, 0C54,05B7,0B74,9847,0106,D ;25936       ALUS?
                                ;25937
                                ;25938
                                ;25939          ;*********************************************************************
                                ;25940          ; Subtraction done, data saved in FLAGS:
                                ;25941          ; FLAG0 = 0 if source 2 is pos., 1 if source 2 is neg.
                                ;25942          ; FLAG1 = 0 if source 1 is pos., 1 if source 1 is neg.
                                ;25943          ; FLAG2 = 0 if !2!=!1!, 1 if !2!<>!1!
                                ;25944          ; FLAG3 = 0 if source 1 is zero, 1 if source 1 is nonzero
                                ;25945          ; Here from ALUS? BUT on BORROW:
                                ;25946          ; =01 is BORROW=0 (CARRY=1), !#2! >= !#1!
                                ;25947          ; =11 is BORROW=1 (CARRY=0), !#2! < !#1!  (subtraction was the ''wrong
                                ;25948          ; way'').
                                ;25949          ;*********************************************************************
                                ;25950 =01
                                ;25951 DS.CMPP.FLAGS:
                                ;25952          ;01-------------------------------;
                                ;25953          R[R0]_0.CLEAR FPD,FLAG<2-0>?,   ;Clear GPR0, FPD_0, split on sub.
U 106D, 04E4,05B7,05B4,0047,0102,0 ;25954       NEXT/DS.CMPP.FLAGS.OK           ;results
                                ;25955
                                ;25956          ;11-------------------------------;
U 106F, 00E4,05B7,05F4,0047,0102,C ;25957       R[R0]_0.CLEAR FPD,FLAG1?        ;Clear GPR0, FPD_0, what's source 1
                                ;25958
                                ;25959          ;*********************************************************************
                                ;25960          ; Subtraction was wrong: if Source 1 was positive, then Src.#1 > Src.#2
                                ;25961          ; regardless of source 2 sign.  If source 1 was negative, then
                                ;25962          ; Src.#1 < Src.#2 for any source 2 sign.  Load CC if necessary (CC
                                ;25963          ; currently is zero (0)), IRD1.
                                ;25964          ;*********************************************************************
                                ;25965
                                ;25966
                                ;25967 =0*
                                ;25968          ;0*-------------------------------;
U 102C, 0080,0036,4130,0047,003F,9 ;25969       IRD1                            ;Src.#1 > Src.#2, CC is already zero,
                                ;25970                                          ;**IRD1**
                                ;25971
                                ;25972          ;1*-------------------------------;
U 102E, 0580,0C37,0130,40A7,003F,9 ;25973       CC_ZLITO[8.],IRD1               ;Src.#1 < Src.#2, load CC, **IRD1**
```

```
                                        ;25974          ;*************************************************************************
                                        ;25975          ; Subtraction was the correct way, here on FLAG<2-0> split, where:
                                        ;25976          ; FLAG3 =0 if source 1 is zero, =1 if nonzero.
                                        ;25977          ; FLAG2 =0 if sources are equal, =1 if not.
                                        ;25978          ; FLAG1 =0 if source 1 is positive, =1 if source 1 is negative.
                                        ;25979          ; FLAG0 =0 if source 2 is positive, =1 if source 2 is negative.
                                        ;25980          ;
                                        ;25981          ; The BUT splits are as follows:
                                        ;25982          ; =000: Sources are equal
                                        ;25983          ; =001: Sources are equal if Source 1 is zero, else src.#1 > src.#2
                                        ;25984          ; =010: Sources are equal if Source 1 is zero, else src.#1 < src.#2
                                        ;25985          ; =011: Sources are equal
                                        ;25986          ; =100: Src.#1 < Src.#2
                                        ;25987          ; =101: Src.#1 > Src.#2
                                        ;25988          ; =110: Src.#1 < Src.#2
                                        ;25989          ; =111: Src.#1 > Src.#2
                                        ;25990          ;*************************************************************************
                                        ;25991 =000
                                        ;25992 DS.CMPP.FLAGS.OK:
                                        ;25993          ;000--------------------------;
U 1020, 0980,0C37,0130,20A7,003F,9      ;25994          CC_ZLIT0[4],IRD1
                                        ;25995
                                        ;25996          ;001--------------------------;
U 1021, 0880,0036,44F0,0047,010C,6      ;25997          FLAG3?,NEXT/DS.CMPP.-0+0        ;CC already has 0 loaded (guess for
                                        ;25998                                          ;Source 1 nonzero case), is source 1
                                        ;25999                                          ;nonzero?
                                        ;26000
                                        ;26001          ;010--------------------------;
U 1022, 0D80,0C37,04F0,40A7,010C,6      ;26002          CC_ZLIT0[8.],FLAG3?,            ;CC_data for source1<>0, did we guess
                                        ;26003          NEXT/DS.CMPP.-0+0               ;correctly?
                                        ;26004
                                        ;26005          ;011--------------------------;
U 1023, 0980,0C37,0130,20A7,003F,9      ;26006          CC_ZLIT0[4],IRD1
                                        ;26007
                                        ;26008          ;100--------------------------;
U 1024, 0580,0C37,0130,40A7,003F,9      ;26009          CC_ZLIT0[8.],IRD1
                                        ;26010
                                        ;26011 DS.CMPP.EXT1:                            ; For CMT066
                                        ;26012          ;       ++++++++++++++++        ; This entry is for DS.CMPP.FLAGS
                                        ;26013          ;101--------------------------;
U 1025, 0080,0036,4130,0047,003F,9      ;26014          IRD1                            ;CC is zero already
                                        ;26015
                                        ;26016          ;110--------------------------;
U 1026, 0580,0C37,0130,40A7,003F,9      ;26017          CC_ZLIT0[8.],IRD1
                                        ;26018
                                        ;26019 DS.CMPP.EXIT:
                                        ;26020          ;111--------------------------;
U 1027, 0080,0036,4130,0047,003F,9      ;26021          IRD1                            ;CC is zero already
```

```
                                    ;26022          ;*************************************************************************
                                    ;26023          ; Here on FLAG3? split: =0 is src.#1=0, =1 is src.#1<>0.  CC is
                                    ;26024          ; set for src.#1<>0 case (either 0000 or 1000).
                                    ;26025          ;*************************************************************************
                                    ;26026 =0
                                    ;26027 DS.CMPP.-0+0:
                                    ;26028          ;0-----------------------------;
U 10C6, 0980,0C37,0130,20A7,003F,9  ;26029          CC_ZLIT0[4],IRD1
                                    ;26030
                                    ;26031          ;1-----------------------------;
U 10C7, 0080,0036,4130,0047,003F,9  ;26032          IRD1
```

```
;26033 .TOC "  Decimal String                    : CMPP4"
;26034
;26035          ;*************************************************************************
;26036          ;
;26037          ; CMPP4 src1len.rw, src1addr.ab, src2len.rw, src2addr.ab
;26038          ;
;26039          ; After initializating, we split on the difference in the byte length
;26040          ; of the sources.  If equal, we branch to CMPP3 code, else we read the
;26041          ; leading bytes of the longer source.  When a nonzero byte is read,
;26042          ; the absolute value of the longer source is known to be greater
;26043          ; than the shorter source and all that remains is to read the
;26044          ; sign byte.  If the leading bytes of the longer source are zero we
;26045          ; patch things up and branch into CMPP3.
;26046          ;
;26047          ; ADDP4 and SUBP4 use this code for preliminary initialization.
;26048          ;
;26049          ;       Input           MDR             Source 1 length
;26050          ;
;26051          ;       Resources       DREG            Misc. data
;26052          ;                       ALUS            BCDSIGN evaluation
;26053          ;                       VA              READs
;26054          ;                       MDR             Returns data
;26055          ;                       PC              Used for XB READs
;26056          ;                       GPR0            Holds Istream PC
;26057          ;                       TEMP1           Source 1 Length
;26058          ;                       TEMP2           Source 1 address
;26059          ;                       TEMP3           Source 2 Length
;26060          ;                       TEMP6           L/2, source 1
;26061          ;                       TEMP7           L/2, source 2
;26062          ;                       FPDOFFSET       _ZLIT0[18.] during seperate
;26063          ;                                       flows (leading byte READs)
;26064          ;                       FLAG0           _0 if src.#2 len. > src.#1 len.
;26065          ;                                       _1 if src.#1 len. > src.#2 len.
;26066          ;
;26067          ;       (If leading bytes =0 or lengths are equal:)
;26068          ;       Output          GPR1            Base add., source 1
;26069          ;                       GPR3            Base add., source 2
;26070          ;                       STEPC           L/2, source 1
;26071          ;                       DREG            Ditto
;26072          ;                       TEMP1           Base address of source 1 to be
;26073          ;                                       used for source READs
;26074          ;                       TEMP2           Base address of source 2 to be
;26075          ;                                       used for source READs
;26076          ;                       MTEMP8          Source Lengths
;26077          ;
;26078          ;       (If leading bytes <>0:)
;26079          ;       Output          GPR0            0
;26080          ;                       GPR1            Base add., source 1
;26081          ;                       GPR2            0
;26082          ;                       GPR3            Base add., source 2
;26083          ;                       CC              Modified
;26084          ;*************************************************************************
```

```
                        ;26085 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;26086 DS.CMPP4:
                        ;26087          ;-------------------------------;
                        ;26088          R[TEMP1]_SEXT(M[MDR]),          ;Save len#1, get add#1
                        ;26089          SIZE[WORD],LOD INC BRA?,
U 03B9, 0485,2E5E,0190,4047,0012,0      ;26090          NEXT/OS.ADD
                        ;26091
                        ;26092 =00
                        ;26093 DS.CMPP4.S2:
                        ;26094          ;00-----------------------------;
                        ;26095          R[TEMP2]_M[MDR],NEXT/OS.RED,    ;Save Add. #1, get len. #2
U 027C, 0085,2592,41B0,8047,0410,0      ;26096          PUSH,LOD INC BRA?
                        ;26097
                        ;26098          ;01-----------------------------;
                        ;26099          R[TEMP3]_SEXT(M[MDR]),          ;TEMP3_Length#2, READ add.#2
                        ;26100          SIZE[WORD],PUSH,
U 027D, 0885,2E5E,0190,C047,0412,0      ;26101          LOD INC BRA?,NEXT/OS.ADD
                        ;26102
                        ;26103          ;10-----------------------------;
U 027E, 0480,1002,6B30,D8E7,010D,A      ;26104          D_M[TEMP1].OR.R[TEMP3],IP.TS?   ;OR lengths for check, ints. pending?
                        ;26105 =
                        ;26106
                        ;26107 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                        ;26108 =0
                        ;26109          ;0------------------------------;
                        ;26110          WB_D.ANDNOT.ZLIT0[31.],         ;Nothing pending, lengths ok?
U 10DA, 0D80,0C33,8A30,F847,010D,E      ;26111          WX.EQ.0?,NEXT/DS.CMPP4.S3
                        ;26112
                        ;26113          ;1------------------------------;
                        ;26114          PUSH,INTPEND OR TIMER?,         ;Something pending, FPD=0, return -1
U 10DB, 0480,0036,4AF0,0047,04F7,0      ;26115          NEXT/IE.SERV.IP.TS2             ;if timer service
                        ;26116
                        ;26117 =0
                        ;26118 DS.CMPP4.S3:
                        ;26119          ;0------------------------------;
U 10DE, 0C80,0036,4030,0047,00FF,8      ;26120          NEXT/IE.OPER.FAULT              ;*Bad, reserved operand fault
                        ;26121
                        ;26122          ;1------------------------------;
U 10DF, 00EC,2592,4034,4047,0101,1      ;26123          R[R1]_M[TEMP2],SET FPD          ;GPR1_Address #1
```

```
                        ;26124          ;---------------------------------;
U 1011, 0885,2592,48B4,C047,010E,8      ;26125          R[R3]_M[MDR],IR<2>?             ;GPR3_Source 2 address, which inst.
                        ;26126                                          ;are we?
                        ;26127
                        ;26128 =0
                        ;26129          ;0---------------------------------;
                        ;26130          M[TEMP9]_R[TEMP3],WB<0>?,       ;ADDP4[20] or SUBP4[22], is dest. odd?
U 10E8, 0486,95BE,42B0,C047,010F,E      ;26131          NEXT/DS.CMPP4.ADD4
                        ;26132
                        ;26133 DS.CMPP4.REE:
                        ;26134          ;1---------------------------------;
                        ;26135          R[TEMP6]_M[TEMP1].SR.1,         ;CMPP4[37], TEMP6_L/2, src.#1
U 10E9, 0084,1591,8031,8047,0103,1      ;26136          NEXT/DS.CMPP4.C1
                        ;26137
                        ;26138
                        ;26139 =0
                        ;26140 DS.CMPP4.ADD4:
                        ;26141          ;0---------------------------------;
                        ;26142          CC_ZLITO[0],                    ;Even dest., PSL<C>_0
U 10FE, 0580,0C37,0030,00A7,0101,A      ;26143          NEXT/DS.CMPP4.ADD4.1
                        ;26144
                        ;26145          ;1---------------------------------;
U 10FF, 0180,0C37,0030,08A7,0101,A      ;26146          CC_ZLITO[1]                     ;Odd dest., PSL<C>_1
                        ;26147
                        ;26148 DS.CMPP4.ADD4.1:
                        ;26149          ;---------------------------------;
U 101A, 0C86,85BE,4030,4047,0102,8      ;26150          M[TEMP8]_R[TEMP1]               ;Copy Source 1 pointer
                        ;26151
                        ;26152          ;---------------------------------;
                        ;26153          R[TEMP8]_M[TEMP9],              ;Source 2 & Dest. lengths are loaded,
U 1028, 0084,9592,4032,0047,0111,2      ;26154          NEXT/DS.ADDP4                   ;continue in <ADDSUB>P4
```

```
                                ;26155 .TOC "  Decimal String                    :       Unequal lengths"
                                ;26156
                                ;26157          ;*************************************************************************
                                ;26158          ; CMPP4 continues here after spliting from <ADDSUB>P4.  If byte lengths
                                ;26159          ; of the sources are equal we split to CMPP3 code.
                                ;26160          ; If the byte lengths are not equal, the leading bytes of the longer
                                ;26161          ; string are read.  If all the leading bytes are zero, we split to
                                ;26162          ; CMPP3, else we read the sign byte of the longer string to determine
                                ;26163          ; which inequality CC to set.
                                ;26164          ;*************************************************************************
                                ;26165
                                ;26166 DS.CMPP4.C1:
                                ;26167          ;-------------------------------;
U 1031, 0884,3591,8031,C047,0103,6  ;26168          R[TEMP7]_M[TEMP3].SR.1        ;TEMP7_L/2, source 2
                                ;26169
                                ;26170          ;-------------------------------;
                                ;26171          STEPC_M[TEMP6]-R[TEMP7],      ;STEPC_byte length differences, split
U 1036, 0080,6000,0B61,C107,0907,8 374*  ;26172          SIGND CMP?,SIZE[LONG]         ;on polarity of result
                                ;26173
                                ;26174 =00
                                ;26175          ;00-----------------------------;
                                ;26176          R[R0]_M[PC],SET FLAG0,        ;(Src.#1 Len. > Src.#2 Len.)
U 1078, 0C45,A592,4034,0047,0104,D  ;26177          NEXT/DS.CMPP4.L1              ;Store Istream PC
                                ;26178
                                ;26179          ;01-----------------------------;
                                ;26180          STEPC_D_M[TEMP1].SR.1,        ;(Src.#1 Len. = Src.#2 Len.)
U 1079, 0880,1001,A03D,8107,0106,0  ;26181          NEXT/DS.CMPP4.34              ;Adjust & enter CMPP3 code
                                ;26182
                                ;26183          ;10-----------------------------;
U 107A, 0405,A592,4034,0047,0103,C  ;26184          R[R0]_M[PC],CLEAR FLAG0       ;(Src.#1 Len. < Src.#2 Len.)
                                ;26185                                        ;Store Istream PC
                                ;26186 =
                                ;26187
                                ;26188          ;-------------------------------;
U 103C, 0880,05BE,4034,C487,0104,6  ;26189          PC_R[R3]                      ;Point XB to base of longer string
                                ;26190
                                ;26191          ;-------------------------------;
                                ;26192          STEPC_M[TEMP7]-R[TEMP6],      ;STEPC_positive difference between
U 1046, 0480,7000,0031,8107,010A,E  ;26193          NEXT/DS.CMPP4.PRELOP          ;lengths, loop through excess
                                ;26194
                                ;26195
                                ;26196 DS.CMPP4.L1:
                                ;26197          ;-------------------------------;
U 104D, 0C80,05BE,4034,4487,010A,E  ;26198          PC_R[R1],NEXT/DS.CMPP4.PRELOP  ;Point XB to base of longer string
```

```
                                ;26199          ;*************************************************************************
                                ;26200          ; Source byte lengths are equal, copy base addresses & enter CMPP3.
                                ;26201          ;*************************************************************************
                                ;26202
                                ;26203          ;++++++++++++
                                ;26204          ;The next two words here hopefully will be in locations
                                ;26205          ;1052 and 1066
                                ;26206          ;++++++++++++
                                ;26207
                                ;26208 QU.RESV.FIX:
                                ;26209          ;+++++++ force address ++++++++;
                                ;26210          ;------------------------------;
                                ;26211          WRITE.UL M[TEMP1].ANDNOT.ZLIT0[1],      ; CLEAR SOFT LOCK BIT
U 1[ _, 0D80,1C13,8020,0DDB,0105,B  ;26212      SIZE[LONG]                              ; AND UNLOCK HARDWARE INTERLOCK
                                ;26213
                                ;26214          ;+++++++ force address ++++++++;
                                ;26215          ;------------------------------;
                                ;26216          R[MM.TEMP3] M[TEMP6],           ; RECOVER ORIGINAL FAULTING ADR
U 105B, 0484,6592,4039,C047,00BF,D  ;26217      NEXT/QU.RESV.EXIT               ; RETURN TO EXCEPTION HANDLER FLOW
                                ;26218
                                ;26219          ;++++++++++++++++++++++++++++++
                                ;26220
                                ;26221 DS.CMPP4.34:
                                ;26222          ;------------------------------;
                                ;26223          M[TEMP2] R[R3],                 ;And source 2 length, enter CMPP3
U 1060, 0486,25BF,4034,C047,0106,B  ;26224      NEXT/DS.CMPP4.FREE2
                                ;26225
                                ;26226
                                ;26227
                                ;26228
                                ;26229          ;*************************************************************************
                                ;26230          ; Source 2 length was longer, the leading bytes read, and all were zero.
                                ;26231          ; Source length_source 1 (shorter) length, Source 1 base address_GPR1,
                                ;26232          ; and adjust Source 2 base address to account for the extra  bytes read.
                                ;26233          ;*************************************************************************
                                ;26234
                                ;26235 DS.CMPP4.FREE2:                          ;This patch frees up two locations
                                ;26236          ;------------------------------;
U 106B, 0486,85BE,4030,4047,0106,E  ;26237      M[TEMP8]_R[TEMP1]               ;Copy source length
                                ;26238
                                ;26239          ;------------------------------;
                                ;26240          M[TEMP1]_R[R1],         ;Duplicate source 1 length
U 106E, 0086,15BE,4034,4047,0100,E  ;26241      NEXT/DS.CMPP3                   ;
                                ;26242
                                ;26243 ;        ;------------------------------;
                                ;26244 ;        M[TEMP8]_R[TEMP1]               ;MTEMP8_source length
                                ;26245 ;
                                ;26246 ;        ;------------------------------;
                                ;26247 ;        M[TEMP1]_R[R1]                  ;Source 1 base address_GPR1
                                ;26248 ;
                                ;26249
                                ;26250 DS.CMPP4.ELOP.2L:
                                ;26251          ;------------------------------;
U 1074, 0880,7000,1031,8047,0107,8  ;26252      Q_M[TEMP7]-R[TEMP6]             ;QREG_difference in bytes
                                ;26253
```

```
                                ;26254          ;---------------------------------;
                                ;26255          M[TEMP2]_R[R3]+Q,                ;Source 2 base address_base+offset,
U 107B, 0086,2039,0034,C047,0106,B ;26256       NEXT/DS.CMPP4.FREE2              ;enter CMPP3 code
                                ;26257
                                ;26258          ;++++++++++++++++++++++++++++++++
                                ;26259          ;THE TWO ROUTINES DS.CMPP4.ELOP.2L AND DS.CMPP4.34
                                ;26260          ;HAVE BEEN REARRANGED TO SAVE 2 LOCATION TO ACCOMODATE
                                ;26261          ;CMT066 CHANGES
                                ;26262          ;++++++++++++++++++++++++++++++++
                                ;26263
                                ;26264
                                ;26265          ;*************************************************************************
                                ;26266          ; Source 1 length was longer, the leading bytes read, and all were zero.
                                ;26267          ; Source length_source 2 (shorter) length, Source 2 base address_GPR3,
                                ;26268          ; and adjust Source 1 base address to account for the extra bytes read.
                                ;26269          ;*************************************************************************
                                ;26270
                                ;26271 DS.CMPP4.ELOP.1L:
                                ;26272
                                ;26273          ;---------------------------------;
U 107E, 0086,85BE,4030,C047,0108,6 ;26274       M[TEMP8]_R[TEMP3]                ;MTEMP8_source length
                                ;26275
                                ;26276          ;---------------------------------;
U 1086, 0C86,25BE,4034,C047,0108,E ;26277       M[TEMP2]_R[R3]                   ;Source 2 base address_GPR3
                                ;26278
                                ;26279          ;---------------------------------;
U 108E, 0880,6000,1031,C047,0109,F ;26280       Q_M[TEMP6]-R[TEMP7]              ;QREG_byte length difference
                                ;26281
                                ;26282          ;---------------------------------;
                                ;26283          M[TEMP1]_R[R1]+Q,                ;Source 1 base address_base+offset,
U 109F, 0486,1039,0034,4047,0100,E ;26284       NEXT/DS.CMPP3                    ;branch into CMPP3 microcode
```

```
                                ;26285          ;*************************************************************************
                                ;26286          ; FLAG0 is clear if source 2 is longer and set if source 1 is longer.
                                ;26287          ; PC points to the base of the longer string.
                                ;26288          ; GPR0 has the Istream PC.
                                ;26289          ;*************************************************************************
                                ;26290
                                ;26291 DS.CMPP4.PRELOP:
                                ;26292          ;-------------------------------;
U 10AE, 0586,CC37,0030,9047,0110,2 ;26293       M[FPDOFFSET]_ZLIT0[18.]         ;CMPP4 (seperate flows) IANDE code
                                ;26294
                                ;26295          ;*************************************************************************
                                ;26296          ; =0 is data is left to READ in string, or first READ.
                                ;26297          ; =1 is STEPC=0, no more data left.
                                ;26298          ;*************************************************************************
                                ;26299 =0
                                ;26300 DS.CMPP4.LOP:
                                ;26301          ;0------------------------------;
                                ;26302          WB_ZEXT(XB) PC_PC+1,            ;READ first/next byte, is it nonzero?
U 1102, 0481,7016,4A4D,A047,0111,A ;26303       WX.NE.0?,NEXT/DS.CMPP4.LOP.SPL
                                ;26304
                                ;26305          ;1------------------------------;
U 1103, 0480,05BE,4434,0487,0111,0 ;26306       PC_R[R0],FLAG0?                 ;Restore PC
                                ;26307
                                ;26308
                                ;26309          ;*************************************************************************
                                ;26310          ; All leading bytes were zero, PC is restored.  =0 is source 2 length
                                ;26311          ; was the longer of the two, so the source 1 length is the amount left
                                ;26312          ; to be read.  =1 is source 1 length was the longer of the two, etc.,
                                ;26313          ; etc.
                                ;26314          ;*************************************************************************
                                ;26315 =0
                                ;26316          ;0------------------------------;
                                ;26317          STEPC_D_M[TEMP1].SR.1,          ;STEPC_DREG_shorter length/2,
U 1110, 0880,1001,A03D,8107,0107,4 ;26318       NEXT/DS.CMPP4.ELOP.2L           ;continue adjusting for CMPP3
                                ;26319
                                ;26320          ;1------------------------------;
                                ;26321          STEPC_D_M[TEMP3].SR.1,          ;Ditto
U 1111, 0C80,3001,A03D,8107,0107,E ;26322       NEXT/DS.CMPP4.ELOP.1L
```

```
                                ;26323          ;*************************************************************************
                                ;26324          ; =0 is data=0, BUT on Step Counter to =0*0 or =0*1 splits at
                                ;26325          ; DS.CMPP4.LOP:.  =1 is data>0, we have a winner.
                                ;26326          ;*************************************************************************
                                ;26327 =0
                                ;26328 DS.CMPP4.LOP.SPL:
                                ;26329          ;0-----------------------------;
U 111A, 0C80,0036,4330,0047,0110,2  ;26330      DBZ STEPC?,NEXT/DS.CMPP4.LOP
                                ;26331
                                ;26332          ;1-----------------------------;
U 111B, 0C80,05BE,4434,0487,0111,E  ;26333      PC_R[R0],FLAG0?                 ;Restore PC, which src. are we reading?
                                ;26334
                                ;26335 =0
                                ;26336          ;0-----------------------------;
                                ;26337          VA_M[TEMP7]+R[R3],              ;Source 2, read sign byte
U 111E, 0480,7001,0034,C4A7,010B,0  ;26338      NEXT/DS.CMPP4.UNEQ.SGN
                                ;26339
                                ;26340          ;1-----------------------------;
U 111F, 0480,6001,0034,44A7,010B,0  ;26341      VA_M[TEMP6]+R[R1]               ;Source 1, read sign byte
                                ;26342
                                ;26343 DS.CMPP4.UNEQ.SGN:
                                ;26344          ;-----------------------------;
U 10B0, 0484,05B7,0004,8050,010C,4  ;26345      READ,SIZE[BYTE],R[R2]_0         ;GPR2_0
```

```
                                ;26346 DS.CMPP4.UNEQ.SPL:
                                ;26347          ;--------------------------------;
                                ;26348          WB_M[MDR],ALUS_BCD SIGN.ZERO,   ;Evaluate the sign, wrap up the inst.,
U 10C4, 04E1,2592,4430,00C7,0112,6 ;26349       FLAG0?,CLEAR FPD                ;"which source are we?"
                                ;26350
                                ;26351 =0
                                ;26352          ;0--------------------------------;
                                ;26353          R[R0]_0,ALUS?,                  ;Src.#2 is larger, GPR0_0
U 1126, 0884,05B7,0B74,1847,010A,9 ;26354       NEXT/DS.CMPP4.UNEQ.CC2
                                ;26355
                                ;26356          ;1--------------------------------;
U 1127, 0084,05C7,0B74,1847,010A,1 ;26357       R[R0]_0,ALUS?                   ;Src.#1 is larger, GPR0_0
                                ;26358
                                ;26359 =01
                                ;26360          ;01-------------------------------;
U 10A1, 0180,0C37,0130,00A7,003F,9 ;26361       CC_ZLITO[0],IRD1                ;|1|>|2|, and source 1 is positive
                                ;26362
                                ;26363          ;11-------------------------------;
U 10A3, 0580,0C37,0130,40A7,003F,9 ;26364       CC_ZLITO[8],IRD1                ;|1|>|2|, and source 1 is negative
                                ;26365
                                ;26366
                                ;26367 =01
                                ;26368 DS.CMPP4.UNEQ.CC2:
                                ;26369          ;01-------------------------------;
U 10A9, 0580,0C37,0130,40A7,003F,9 ;26370       CC_ZLITO[8],IRD1                ;|1|<|2|, and source 2 is positive
                                ;26371
                                ;26372          ;11-------------------------------;
U 10AB, 0180,0C37,0130,00A7,003F,9 ;26373       CC_ZLITO[0],IRD1                ;|1|<|2|, and source 2 is negative
```

```
;26374 .TOC "  Decimal String                    : SUBP6 and ADDP6"
;26375
;26376          ;*************************************************************************
;26377          ;
;26378          ;       SUBP6 sublen.rw, subaddr.ab, minlen.rw, minaddr.ab, diflen.rw,
;26379          ;             difaddr.ab
;26380          ;
;26381          ;       ADDP6 add1len.rw, add1addr.ab, add2len.rw, add2addr.ab,
;26382          ;             sumlen.rw, sumaddr.ab
;26383          ;
;26384          ;       MULP and DIVP perform operand fetch and preliminary
;26385          ;       initialization in this code.
;26386          ;
;26387          ;       Instructions initialize and branch into ADDP4 code.
;26388          ;
;26389          ;       Input           MDR             add1len.rw
;26390          ;
;26391          ;       Resources       TEMP1           Misc. data
;26392          ;                       TEMP2           Ditto
;26393          ;                       TEMP3           Ditto
;26394          ;                       TEMP4           Ditto
;26395          ;                       TEMP5           Ditto
;26396          ;                       DREG            ORs Lengths
;26397          ;                       MDR             READs
;26398          ;
;26399          ;       Output          GPR1            Base address, source 1
;26400          ;                       GPR3            Base address, source 2
;26401          ;                       GPR4            0
;26402          ;                       GPR5            Base address, destination
;26403          ;                       MTEMP8          Source 1 Length
;26404          ;                       MTEMP9          Source 2 Length
;26405          ;                       RTEMP8          Destination Length
;26406          ;                       FPD             Set
;26407          ;                       CC              _0 if destination is even,
;26408          ;                                       _1 if destination is odd
;26409          ;                       Lengths         Checked for Reserved Operand
;26410          ;                                       Fault
;26411          ;
;26412          ;*************************************************************************
```

```
                        ;26413 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;26414 =00
                        ;26415 DS.ADDP6:
                        ;26416          ;00-------------------------------;
                        ;26417          R[TEMP1]_SEXT(M[MDR]),          ;Hold length #1, get addr. #1,
                        ;26418          NEXT/OS.ADD,LOD INC BRA?,       ;PUSH for RETURN from 5th OSR
U 0288, 0C85,2E5E,0190,4047,0412,0      ;26419          PUSH,SIZE[WORD]                 ;routine.  Next ADDP6 microword
                        ;26420                                          ;is DS.ADDP6.2:
                        ;26421
                        ;26422          ;01-------------------------------;
                        ;26423          R[TEMP5]_SEXT(M[MDR]),          ;TEMP5_Dest. Length, READ dest.
                        ;26424          PUSH,LOD INC BRA?,NEXT/OS.ADD,  ;address
U 0289, 0885,2E5E,0191,4047,0412,0      ;26425          SIZE[WORD]
                        ;26426
                        ;26427          ;10-------------------------------;
                        ;26428          D_D.OR.R[TEMP5],                ;MDR has dest. add., OR it in
U 028A, 0080,0022,6031,4047,010C,E      ;26429          NEXT/DS.ADDP6.LEN
                        ;26430 =
                        ;26431
                        ;26432 =00
                        ;26433 DS.ADDP6.2:
                        ;26434          ;00-------------------------------;
                        ;26435          R[TEMP2]_M[MDR],PUSH,           ;Hold addr#1, get length #2
U 0294, 0085,2592,41B0,8047,0410,0      ;26436          NEXT/OS.RED,LOD INC BRA?
                        ;26437
                        ;26438          ;01-------------------------------;
                        ;26439          R[TEMP3]_SEXT(M[MDR]),          ;Hold length#2, get addr.#2
                        ;26440          PUSH,NEXT/OS.ADD,LOD INC BRA?,
U 0295, 0885,2E5E,0190,C047,0412,0      ;26441          SIZE[WORD]
                        ;26442
                        ;26443          ;10-------------------------------;
U 0296, 0C85,2592,4B31,18E7,0120,8      ;26444          R[TEMP4]_M[MDR],IP.TS?          ;TEMP4_Source 2 address, anything
                        ;26445                                          ;pending?
                        ;26446 =
                        ;26447
                        ;26448 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                        ;26449 =0
                        ;26450          ;0-------------------------------;
                        ;26451          D_M[TEMP1].OR.R[TEMP3],         ;D_Source Length OR, READ dest. length,
U 1208, 0880,1002,61B0,C047,0010,0      ;26452          LOD INC BRA?,NEXT/OS.RED        ;RETURN TO DS.ADDP6:+1
                        ;26453
                        ;26454          ;1-------------------------------;
                        ;26455          INTPEND OR TIMER?,PUSH,         ;RETURN-1 or roll back
U 1209, 0480,0036,4AF0,0047,04F7,0      ;26456          NEXT/IE.SERV.IP.TS2
```

```
                                ;26457          ;*****************************************************************************
                                ;26458          ; Operand fetches done, check lengths for legality, do preliminary
                                ;26459          ; init.
                                ;26460          ;*****************************************************************************
                                ;26461
                                ;26462 DS.ADDP6.LEN:
                                ;26463          ;-------------------------------;
                                ;26464          WB_D.ANDNOT.ZLIT0[31.],         ;Are lengths ok?
U 10CE, 0580,0C33,8A30,F847,0120,A ;26465       WX.EQ.0?
                                ;26466
                                ;26467 =0
                                ;26468          ;0------------------------------;
U 120A, 0C80,0036,4030,0047,00FF,8 ;26469       NEXT/IE.OPER.FAULT
                                ;26470
                                ;26471          ;1------------------------------;
U 120B, 00EC,2592,4034,4047,010D,1 ;26472       R[R1]_M[TEMP2],SET FPD          ;Source 1 address
                                ;26473
                                ;26474          ;-------------------------------;
U 10D1, 0084,4592,4674,C047,0102,9 ;26475       R[R3]_M[TEMP4],IR<2-0>?         ;Source 2 address, which inst. are we?
                                ;26476
                                ;26477 =001
                                ;26478
                                ;26479          ;001------------------------------;
U 1029, 0085,2592,4035,4047,010D,6 ;26480       R[R5]_M[MDR],NEXT/DS.ADDP6.EXIT ;ADDP6[21], GPR5_dest. add.
                                ;26481
                                ;26482          ;011------------------------------;
U 102B, 0085,2592,4035,4047,010D,6 ;26483       R[R5]_M[MDR],NEXT/DS.ADDP6.EXIT ;SUBP6[23], GPR5_dest. add.
                                ;26484
                                ;26485          ;101------------------------------;
U 102D, 0085,2592,4035,4047,010D,6 ;26486       R[R5]_M[MDR],NEXT/DS.ADDP6.EXIT ;MULP[25], GPR5_dest. add.
                                ;26487
                                ;26488          ;111------------------------------;
U 102F, 0885,2592,4035,4047,011E,C ;26489       R[R5]_M[MDR],NEXT/DS.DIVP       ;DIVP[27], ditto+continue in DIVP code
```

```
                                ;26490          ;*****************************************************************************
                                ;26491          ; ADDP6, SUBP6, and MULP continue initialization
                                ;26492          ;*****************************************************************************
                                ;26493
                                ;26494 DS.ADDP6.EXIT:
                                ;26495          ;--------------------------------;
U 10D6, 0084,5592,42B2,0047,0120,C ;26496       R[TEMP8]_M[TEMP5],WB<0>?        ;Dest. Length, is it odd?
                                ;26497
                                ;26498 =0
                                ;26499          ;0-------------------------------;
                                ;26500          CC_ZLITO[0],                    ;No, even, PSL<C>_0
U 120C, 0580,0C37,0030,00A7,010D,C ;26501       NEXT/DS.ADDP6.EXIT.2
                                ;26502
                                ;26503          ;1-------------------------------;
U 120D, 0180,0C37,0030,08A7,010D,C ;26504       CC_ZLITO[1]                     ;Yes, PSL<C>_1
                                ;26505
                                ;26506
                                ;26507 DS.ADDP6.EXIT.2:
                                ;26508          ;--------------------------------;
U 10DC, 0C86,85BE,4030,4047,010E,4 ;26509       M[TEMP8]_R[TEMP1]               ;Source 1 Length
                                ;26510
                                ;26511          ;--------------------------------;
U 10E4, 0C84,05B7,0035,0047,0110,B ;26512       R[R4]_0                         ;GPR4_0
                                ;26513
                                ;26514          ;--------------------------------;
                                ;26515          M[TEMP9]_R[TEMP3],              ;Source 2 Length, rest of work is in
U 110B, 0486,95BE,4030,C047,0111,2 ;26516       NEXT/DS.ADDP4                   ;ADDP4 code
```

```
;26517 .TOC "  Decimal String                    : ADDP4 and SUBP4"
;26518
;26519          ;*****************************************************************************
;26520          ;
;26521          ;       SUBP4 sublen.rw, subaddr.ab, diflen.rw, difaddr.ab
;26522          ;       ADDP4 addlen.rw, addaddr.ab, sumlen.rw, sumaddr.ab
;26523          ;
;26524          ;       MULP initializes in ADDP6/SUBP6 code, continues here,
;26525          ;       performs majority of work in MULP code, and returns to ADDP4
;26526          ;       code to set CC & convert -0 to +0.
;26527          ;
;26528          ;       SUBP4 and ADDP4 perform operand fetch and preliminary
;26529          ;       initialization in CMPP4 code.
;26530          ;
;26531          ;       SUBP6 and ADDP6 do their own stuff and branch to ADDP4 code.
;26532          ;       All 5 instructions enter at DS.ADDP4:
;26533          ;
;26534          ;       Input           (& means only for ADDP6, SUBP6, or MULP):
;26535          ;                       GPR1            Source 1 address
;26536          ;                       GPR3            Source 2 address
;26537          ;                       GPR4            Zero (&)
;26538          ;                       GPR5            Destination address (&)
;26539          ;                       MTEMP8          Source 1 Length
;26540          ;                       MTEMP9          Source 2 Length
;26541          ;                       RTEMP8          Destination Length
;26542          ;                                       (for ADDP4 & SUBP4 this is
;26543          ;                                       equal to Source 2 Length)
;26544          ;                       FPD             =1
;26545          ;                       CC              =0 if Dest. is even
;26546          ;                                       =1 if Dest. is odd
;26547          ;
;26548          ;       Resources       <Above, plus:>
;26549          ;                       TEMP2           Misc. Data
;26550          ;                       TEMP3-TEMP6     Source 2 holder
;26551          ;                       RTEMP10-RTEMP13 Source 1 holder
;26552          ;                       FPDOFFSET       ZLIT0[14.]
;26553          ;                       STEPC           I/O routine counter
;26554          ;                       WDR             WRITEs
;26555          ;                       MDR             READs
;26556          ;                       VA              I/O address
;26557          ;                       ALUS            overflow & sign checking
;26558          ;                       DREG            Misc. data
;26559          ;                       QREG            Ditto
```

```
                    ;26560          ;                       FLAG0           1) Source 2 sign
                    ;26561          ;                                       2) DS.WRITE temp.
                    ;26562          ;                                       3) masks <C>
                    ;26563          ;                       FLAG1           1) Source 1 sign
                    ;26564          ;                                       2) <V>image, 0 before WRITE
                    ;26565          ;                       FLAG2           <Z>image, initially 1
                    ;26566          ;                       FLAG3           <N>image
                    ;26567          ;
                    ;26568          ;       Output          GPR0            0
                    ;26569          ;                       GPR1            Base address, source 1
                    ;26570          ;                       GPR2            0
                    ;26571          ;                       GPR3            Base address, destination
                    ;26572          ;                       PSL<3:0>        Set/cleared, etc.
                    ;26573          ;
                    ;26574          ;*******************************************************************************
```

```
                                ;26575 DS.ADDP4:
                                ;26576 DS.ADDP4.REE:
                                ;26577          ;--------------------------------;
U 1112, 0080,8001,A03D,8107,0111,7 ;26578       STEPC_D_M[TEMP8].SR.1           ;STEPC_D_L/2 for source 1 read
                                ;26579
                                ;26580          ;--------------------------------;
                                ;26581          M[FPDOFFSET]_ZLITO[14.],        ;Pack routine code, anything pending?
                                ;26582          COUNT OR INT TIMER?,SIZE[LONG],
U 1117, 0586,CC37,0B20,7047,090B,2 350* ;26583  NEXT/DS.ADDP4.SRC1
                                ;26584
                                ;26585 =00
                                ;26586 =01
                                ;26587          ;01--------------------------------;
                                ;26588          STEPC_D_M[TEMP9].SR.1,          ;STEPC_D_L/2 for source #2,
U 10B1, 0C80,9001,A8BD,8107,0120,E ;26589       IR<2>?,NEXT/DS.ADDP4.OPC        ;which inst. are we?
                                ;26590
                                ;26591 DS.ADDP4.SRC1:
                                ;26592          ;10--------------------------------;
U 10B2, 0C80,00A1,0034,44A7,0505,D ;26593       VA_D+R[R1]+1,PUSH,NEXT/DS.READ  ;No ints, or timer service, read
                                ;26594                                          ;source 1, RETURN-1
                                ;26595
                                ;26596          ;11--------------------------------;
                                ;26597          INTPEND OR TIMER?,PUSH,         ;Yes, RETURN-1 or to IE.PACK.DONE
U 10B3, 0480,0036,4AF0,0047,04F7,0 ;26598       NEXT/IE.SERV.IP.TS2
                                ;26599
                                ;26600 =0
                                ;26601 DS.ADDP4.OPC:
                                ;26602          ;0---------------------------------;
                                ;26603          R[TEMP10]_M[TEMP3],ALUS?,       ;ADDPx, SUBPx, begin moving source 1,
U 120E, 0484,3592,4B72,9847,010B,9 ;26604       NEXT/DS.ADDP4.SRC1SGN           ;what is sign?
                                ;26605
                                ;26606          ;1---------------------------------;
                                ;26607          R[R0]_M[TEMP8].RR.24,           ;MULP[25], mask in src.#1 length
U 120F, 0884,83B7,0024,0047,0118,3 ;26608       NEXT/DS.MULP
                                ;26609
                                ;26610
                                ;26611 =01
                                ;26612 DS.ADDP4.SRC1SGN:
                                ;26613          ;01--------------------------------;
                                ;26614          CLEAR FLAG1,                    ;Positive
                                ;26615          R[TEMP11]_M[TEMP4],
U 10B9, 0C0C,4592,4032,C047,0111,C ;26616       NEXT/DS.ADDP4.SRC2
                                ;26617
                                ;26618          ;11--------------------------------;
                                ;26619          SET FLAG1,                      ;Negative
U 10BB, 044C,4592,4032,C047,0111,C ;26620       R[TEMP11]_M[TEMP4]
```

```
                                ;26621 DS.ADDP4.SRC2:
                                ;26622          ;-------------------------------;
U 111C, 0484,5592,4033,0047,0112,B ;26623       R[TEMP12]_M[TEMP5]
                                ;26624
                                ;26625          ;-------------------------------;
                                ;26626          R[TEMP13]_M[TEMP6],             ;Now read source 2
U 112B, 0884,6592,4033,4047,0112,A ;26627       NEXT/DS.ADDP4.REWR
                                ;26628
                                ;26629
                                ;26630          ;*********************************************************************
                                ;26631          ; Source 2 read at =10 will RETURN-1 to =01.  PUSH is made at
                                ;26632          ; =01 for RETURN-1 from future DS.WRITE call, and will RETURN to =00.
                                ;26633          ;*********************************************************************
                                ;26634 =00
                                ;26635          ;00-------------------------------;
                                ;26636          D_R[TEMP8].SR.1,FLAG<2-0>?,     ;D_offset for dest. sign byte,
U 1128, 0080,05BD,A5B2,0047,0105,8 ;26637       NEXT/DS.ADDP4.FLAGS             ;split on <ZVC>image
                                ;26638
                                ;26639          ;01-------------------------------;
                                ;26640          PUSH,NEXT/DS.ADDP4.SRC2SIGN,    ;STEPC_Q_L/2 for DS.WRITE, what is
U 1129, 0880,0E7E,9B72,1907,050C,D ;26641       STEPC_Q_R[TEMP8].SR.1,ALUS?     ;source 2 sign?
                                ;26642
                                ;26643 DS.ADDP4.REWR:
                                ;26644          ;10-------------------------------;
U 112A, 0880,00A1,0034,C4A7,0505,D ;26645       VA_D+R[R3]+1,PUSH,NEXT/DS.READ
                                ;26646 =
                                ;26647
                                ;26648
                                ;26649 =01
                                ;26650 DS.ADDP4.SRC2SIGN:
                                ;26651          ;01-------------------------------;
                                ;26652          CLEAR FLAGO,IR<2-0>?,           ;Source 2 is positive,
U 10CD, 0800,0036,4670,0047,0113,0 ;26653       NEXT/DS.ADDP4.OPCODE            ;which inst. are we?
                                ;26654
                                ;26655          ;11-------------------------------;
U 10CF, 0040,0036,4670,0047,0113,0 ;26656       SET FLAGO,IR<2-0>?              ;Source 2 is negative, ditto
```

```
                                ;26657          ;*************************************************************************
                                ;26658          ; Sources are READ, signs remembered in FLAG<1-0>, QREG & STEPC are
                                ;26659          ; loaded for DS.WRITE, here on IR<2-0>? BUT for VA load & decision
                                ;26660          ; on whether to add or subtract.  All 4 BUT splits BUT on old value of
                                ;26661          ; FLAG<1-0> to split on source signs.
                                ;26662          ; =*00 is ADDP4[20]
                                ;26663          ; =*01 is ADDP6[21]
                                ;26664          ; =*10 is SUBP4[22]
                                ;26665          ; =*11 is SUBP6[23]
                                ;26666          ;*************************************************************************
                                ;26667 =*00
                                ;26668 DS.ADDP4.OPCODE:
                                ;26669          ;*00--------------------------;
                                ;26670          VA_R[R3],
U 1130, 0880,05BE,4534,C4A7,0113,4 ;26671       NEXT/DS.ADDP4.ADD.SPL,FLAG<1-0>?
                                ;26672
                                ;26673          ;*01--------------------------;
                                ;26674          VA_R[R5],
U 1131, 0880,05BE,4535,44A7,0113,4 ;26675       NEXT/DS.ADDP4.ADD.SPL,FLAG<1-0>?
                                ;26676
                                ;26677          ;*10--------------------------;
                                ;26678          VA_R[R3],
U 1132, 0880,05BE,4534,C4A7,0113,8 ;26679       NEXT/DS.ADDP4.SUB.SPL,FLAG<1-0>?
                                ;26680
                                ;26681          ;*11--------------------------;
                                ;26682          VA_R[R5],
U 1133, 0880,05BE,4535,44A7,0113,8 ;26683       NEXT/DS.ADDP4.SUB.SPL,FLAG<1-0>?
                                ;26684 =
```

```
                                ;26685          ;*************************************************************************
                                ;26686          ; ADDP<46> inst, here on FLAG<1-0> BUT.  Assumption is made that
                                ;26687          ; !src.#2! > !src.#1!, and destination sign is guessed to be the
                                ;26688          ; sign of source 2.  FLAG3_0 if destination is expected to be positive,
                                ;26689          ; FLAG3_1 if negative.
                                ;26690          ; =00 is +2 + +1, do addition.
                                ;26691          ; =01 is -2 + +1, do subtraction.
                                ;26692          ; =10 is +2 + -1, do subtraction.
                                ;26693          ; =11 is -2 + -1, do addition.
                                ;26694          ;*************************************************************************
                                ;26695 =00
                                ;26696 DS.ADDP4.ADD.SPL:
                                ;26697          :00-----------------------------;
                                ;26698          CLEAR FLAG3,                    ;Dest. is positive
                                ;26699          M[TEMP3]_(MB+R[TEMP10]).BCD,
U 1134, 0C1E,3001,4022,8047,0113,F ;26700       NEXT/DS.ADDP4.ADD
                                ;26701
                                ;26702          ;01-----------------------------;
                                ;26703          SET FLAG3,                      ;Dest. is negative
                                ;26704          M[TEMP3]_(MB-R[TEMP10]).BCD,
U 1135, 005E,3000,4022,8047,0115,0 ;26705       NEXT/DS.ADDP4.SUB
                                ;26706
                                ;26707          ;10-----------------------------;
                                ;26708          CLEAR FLAG3,                    ;Dest. is pos.
                                ;26709          M[TEMP3]_(MB-R[TEMP10]).BCD,
U 1136, 081E,3000,4022,8047,0115,0 ;26710       NEXT/DS.ADDP4.SUB
                                ;26711
                                ;26712          ;11-----------------------------;
                                ;26713          SET FLAG3,                      ;Dest. is neg.
                                ;26714          M[TEMP3]_(MB+R[TEMP10]).BCD,
U 1137, 045E,3001,4022,8047,0113,F ;26715       NEXT/DS.ADDP4.ADD
```

```
                                ;26716          ;*************************************************************************
                                ;26717          ; SUBP<46> inst, here on FLAG<1-0> BUT.  Assumption is made that
                                ;26718          ; |src.#2| > |src.#1|, and destination sign is guessed to be the
                                ;26719          ; sign of source 2.  FLAG3_0 if destination is expected to be positive,
                                ;26720          ; FLAG3_1 if negative.
                                ;26721          ; =00 is +2 - +1, subtract sources.
                                ;26722          ; =01 is -2 - +1, add sources.
                                ;26723          ; =10 is +2 - -1, add.
                                ;26724          ; =11 is -2 - -1, subtract.
                                ;26725          ;*************************************************************************
                                ;26726 =00
                                ;26727 DS.ADDP4.SUB.SPL:
                                ;26728          ;00-----------------------------;
                                ;26729          CLEAR FLAG3,                    ;Dest. is positive
                                ;26730          M[TEMP3]_(MB-R[TEMP10]).BCD,
U 1138, 081E,3000,4022,8047,0115,0 ;26731       NEXT/DS.ADDP4.SUB
                                ;26732
                                ;26733          ;01-----------------------------;
                                ;26734          SET FLAG3,                      ;Dest. is negative
                                ;26735          M[TEMP3]_(MB+R[TEMP10]).BCD,
U 1139, 045E,3001,4022,8047,0113,F ;26736       NEXT/DS.ADDP4.ADD
                                ;26737
                                ;26738          ;10-----------------------------;
                                ;26739          CLEAR FLAG3,                    ;Dest. is pos.
                                ;26740          M[TEMP3]_(MB+R[TEMP10]).BCD,
U 113A, 0C1E,3001,4022,8047,0113,F ;26741       NEXT/DS.ADDP4.ADD
                                ;26742
                                ;26743          ;11-----------------------------;
                                ;26744          SET FLAG3,                      ;Dest. is neg.
                                ;26745          M[TEMP3]_(MB-R[TEMP10]).BCD,
U 113B, 005E,3000,4022,8047,0115,0 ;26746       NEXT/DS.ADDP4.SUB
```

```
                                ;26747 .TOC     "       Decimal String                  :       Addition"
                                ;26748
                                ;26749          ;*************************************************************************
                                ;26750          ; Finish adding sources, split on destination sign & mask in
                                ;26751          ; preferred sign.  Branch is made to DS.WRITE due to PUSH made
                                ;26752          ; after source 2 READ.  Longword carry is made via ALK<C>.
                                ;26753          ;*************************************************************************
                                ;26754
                                ;26755 DS.ADDP4.ADD:
                                ;26756          ;-------------------------------;
U 113F, 0886,4041,4022,C047,0114,3  ;26757          M[TEMP4]_(MB+R[TEMP11]+ALKC).BCD
                                ;26758
                                ;26759          ;-------------------------------;
U 1143, 0086,5041,4023,0047,0114,B  ;26760          M[TEMP5]_(MB+R[TEMP12]+ALKC).BCD
                                ;26761
                                ;26762          ;-------------------------------;
                                ;26763          M[TEMP6]_(MB+R[TEMP13]+ALKC).BCD,
U 114B, 0886,6041,4023,50C7,0114,E  ;26764          ALUS_UNSGN
                                ;26765
                                ;26766          ;-------------------------------;
U 114E, 0480,0036,4B70,1847,010D,5  ;26767          ALUS?                           ;Was there overflow?
                                ;26768
                                ;26769 =01
                                ;26770          ;01-----------------------------;
                                ;26771          SET FLAG1,FLAG3?,               ;Yes, <V>image_1, what is dest. sign?
U 10D5, 0048,0036,44F0,0047,0121,0  ;26772          NEXT/DS.ADDP4.WRITE
                                ;26773
                                ;26774          ;11-----------------------------;
U 10D7, 0808,0036,44F0,0047,0121,0  ;26775          CLEAR FLAG1,FLAG3?              ;No, <V>image_0, what is dest. sign?
                                ;26776
                                ;26777
                                ;26778
                                ;26779          ;*************************************************************************
                                ;26780          ; Here from addition or good subtraction.  <V>image is 0 (ok) or
                                ;26781          ; 1 (ov already from carry out).  PUSH for DS.WRITE was made after
                                ;26782          ; return from reading source 2 & we RETURN+1 to there.
                                ;26783          ; =0 is FLAG3=0, dest. is positive, init. <Z>image.
                                ;26784          ; =1 is FLAG3=1, dest. is negative, init. <Z>image.
                                ;26785          ;*************************************************************************
                                ;26786 =0
                                ;26787 DS.ADDP4.WRITE:
                                ;26788          ;0-----------------------------;
                                ;26789          M[TEMP3]_MB.OR.ZLIT24[0C],
U 1210, 0556,3C92,4030,6047,012C,E  ;26790          NEXT/DS.WRITE,SET FLAG2
                                ;26791
                                ;26792          ;1-----------------------------;
                                ;26793          M[TEMP3]_MB.OR.ZLIT24[0D],
U 1211, 0156,3C92,4030,6847,012C,E  ;26794          NEXT/DS.WRITE,SET FLAG2
```

```
                                ;26795 .TOC     "       Decimal String                  :       Subtraction"
                                ;26796
                                ;26797          ;*************************************************************************
                                ;26798          ; Finish subtraction, with sign of destination the same as source 2
                                ;26799          ; due to guess that |src.#2| > |src.#1|.  If a BORROW state exists
                                ;26800          ; after last subtract, subtraction was made the wrong way (i.e.
                                ;26801          ; |src.#2| < |src.#1|) and a 0-dest. subtraction is made to correct
                                ;26802          ; the situation.  Also, in such a case the destination sign & <N>image
                                ;26803          ; are reversed.
                                ;26804          ;*************************************************************************
                                ;26805
                                ;26806 DS.ADDP4.SUB:
                                ;26807          ;-------------------------------;
U 1150, 0C86,4040,4022,C047,0115,2      ;26808          M[TEMP4]_(MB-R[TEMP11]-ALKC).BCD;BORROW "from" last subtract
                                ;26809
                                ;26810          ;-------------------- ----------;
U 1152, 0C86,5040,4023,0047,0115,7      ;26811          M[TEMP5]_(MB-R[TEMP12]-ALKC).BCD
                                ;26812
                                ;26813          ;-------------------------------;
                                ;26814          M[TEMP6]_(MB-R[TEMP13]-ALKC).BCD,       ;Save BORROW state in ALUS
U 1157, 0486,6040,4023,50C7,0116,4      ;26815          ALUS_UNSGN
                                ;26816
                                ;26817
                                ;26818          ;-------------------------------;
U 1164, 000C,05B7,0B70,9847,010E,5      ;26819          R[TEMP2]_0,ALUS?.CLEAR FLAG1    ;TEMP2_0 for 0-dest. correction,
                                ;26820                                          ;<V>image_0, was sub ok?
                                ;26821
                                ;26822 =01
                                ;26823          ;01-------------------------------;
U 10E5, 0880,0036,44F0,0047,0121,0      ;26824          F ^G3?,NEXT/DS.ADDP4.WRITE      ;Yes, split on dest. sign
                                ;26825
                                ;26826          ;11-------------------------------;
U 10E7, 0084,2000,4020,C047,0116,6      ;26827          R[TEMP3]_(M[TEMP2]-RB).BCD      ;No, begin correction
```

```
                                ;26828 .TOC   "      Decimal String                    :                Correction"
                                ;26829
                                ;26830        ;*************************************************************************
                                ;26831        ; Finish correcting dest., reverse signs, and branch to DS.WRITE.
                                ;26832        ;*************************************************************************
                                ;26833
                                ;26834        ;-------------------------------;
U 1166, 0084,2040,4021,0047,0117,1  ;26835        R[TEMP4]_(M[TEMP2]-RB-ALKC).BCD ;Carry BORROW
                                ;26836
                                ;26837        ;-------------------------------;
U 1171, 0484,2040,4021,4047,0117,2  ;26838        R[TEMP5]_(M[TEMP2]-RB-ALKC).BCD
                                ;26839
                                ;26840        ;-------------------------------;
                                ;26841        R[TEMP6]_(M[TEMP2]-RB-ALKC).BCD,;Init. <2>image to 1
U 1172, 0854,2040,44E1,8047,0121,2  ;26842        FLAG3?,SET FLAG2
                                ;26843
                                ;26844 =0
                                ;26845        ;0-----------------------------;
                                ;26846        SET FLAG3,                      ;Dest. was positive, is now neg.,
                                ;26847        M[TEMP3]_MB.OR.ZLIT24[0D],      ;mask in correct sign
U 1212, 095E,3C92,4030,6847,012C,E  ;26848        NEXT/DS.WRITE
                                ;26849
                                ;26850        ;1-----------------------------;
                                ;26851        CLEAR FLAG3,                    ;Dest. was neg., is now pos., mask in
                                ;26852        M[TEMP3]_MB.OR.ZLIT24[0C],      ;pos. sign
U 1213, 051E,3C92,4030,6047,012C,E  ;26853        NEXT/DS.WRITE
```

```
                                ;26854 .TOC "  Decimal String                    :       CC setting"
                                ;26855
                                ;26856          ;*************************************************************************
                                ;26857          ; <ADDSUB>Px and MULP enter here with DREG containing dest.L/2 and
                                ;26858          ; utilize overlapping BUTs:
                                ;26859          ; 1) From DS.WRITE RETURN:
                                ;26860          ;                               =000 if dest. is nonzero, no OV
                                ;26861          ;                               =010 if dest. is nonzero, OV
                                ;26862          ;                               =100 if dest. is zero, no OV
                                ;26863          ;                               =110 if dest. is zero, OV
                                ;26864          ; 3) From =100 split below:     Dest. is zero, no OV, split to:
                                ;26865          ;                               =000 if dest. is positive
                                ;26866          ;                               =001 if dest. is negative
                                ;26867          ;*************************************************************************
                                ;26868 =000
                                ;26869 DS.ADDP4.FLAGS:
                                ;26870          ;000--------------------------;
U 1058, 0884,0036,4034,8387,0138,E ;26871       R[R2]_FLAGS,NEXT/DS.CCPCIRD1    ;CC is set, finish inst. in ASHP code
                                ;26872
                                ;26873          ;001--------------------------;
                                ;26874          CLEAR FLAG3,IR<2-0>?,           ;<N>image_0, guess xxxP<4> for VA
U 1059, 0018,0021,0674,C4A7,0101,E ;26875       NEXT/DS.ADDP4.FLAGS.SIGN,       ;pointing, which inst. are we really?
                                ;26876          VA_D+R[R3]
                                ;26877
                                ;26878
                                ;26879          ;010--------------------------;
U 105A, 0884,0036,4034,8387,0138,E ;26880       R[R2]_FLAGS,NEXT/DS.CCPCIRD1    ;CC is ok, finish inst. in ASHP code
                                ;26881
                                ;26882 =100
                                ;26883          ;100--------------------------;
U 105C, 0080,0036,44F0,0047,0105,8 ;26884       FLAG3?,NEXT/DS.ADDP4.FLAGS      ;What is sign of dest.?
                                ;26885
                                ;26886 =110
                                ;26887          ;110--------------------------;
U 105E, 0C18,0036,4030,0047,0105,8 ;26888       CLEAR FLAG3,NEXT/DS.ADDP4.FLAGS ;Ensure CC_0110, sign nibble is what
                                ;26889                                          ;ever it is
                                ;26890 =
                                ;26891
                                ;26892 =110
                                ;26893 DS.ADDP4.FLAGS.SIGN:
                                ;26894          ;110--------------------------;
                                ;26895          WRITE ZLITO[0C],SIZE[BYTE],     ;VA is ok, dest._+0, finish inst.
U 101E, 0180,0C37,0000,65D8,0105,8 ;26896       NEXT/DS.ADDP4.FLAGS
                                ;26897
                                ;26898          ;111--------------------------;
                                ;26899          VA_D+R[R5],                     ;MULP[25],ADDP6[21],SUBP6[23], point VA
U 101F, 0C80,0021,0035,44A7,0101,E ;26900       NEXT/DS.ADDP4.FLAGS.SIGN        ;to correct byte
```

```
;26901 .TOC "  Decimal String                    : MULP"
;26902
;26903          ;*************************************************************************
;26904          ;
;26905          ;       MULP mulrlen.rw, mulraddr.ab, muldlen.rw, muldaddr.ab,
;26906          ;               prodlen.rw, prodaddr.ab
;26907          ;
;26908          ;       MULP begins in <ADDSUB>P6 code, continues in <ADDSUB>P4 code
;26909          ;       and enters at DS.MULP.  CC setting and -0 to +0 conversion is
;26910          ;       also done in <ADDSUB>P4 code.
;26911          ;
;26912          ;       The destination is created in MTEMP8-MERRCOD using the
;26913          ;       double-and-divide method for multiplication of longwords with
;26914          ;       overlapping adds to destination TEMPs.  Source 2 is the
;26915          ;       "outer loop", Source 1 the "inner loop".  To speed things up
;26916          ;       an outer loop pass is skipped if the source 2 longword is zero,
;26917          ;       and an inner loop pass is skipped if the source 1 multiplier
;26918          ;       longword is zero.
;26919          ;
;26920          ;       Input           GPR0            MTEMP8 in byte#1
;26921          ;                       GPR1            Source 1 address
;26922          ;                       GPR3            Source 2 address
;26923          ;                       GPR4            Zero
;26924          ;                       GPR5            Destination address
;26925          ;                       TEMP3-TEMP6     Source 1, memory format, sign
;26926          ;                                       nibble zeroed, LSB in byte#3
;26927          ;                                       of TEMP3, MSB in byte#0 of
;26928          ;                                       TEMP6
;26929          ;               RTEMP8          Destination Length
;26930          ;               STEPC           L/2, source #2
;26931          ;               DREG            L/2, source #2
;26932          ;               CC              =0 if Dest. is even
;26933          ;                               =1 if Dest. is odd
;26934          ;               ALUS            BCDSIGN data
;26935          ;               Lengths         Checked for Reserved Operand
;26936          ;                               Fault
;26937          ;               FPD             =1
```

```
;26938
;26939
;26940
;26941
;26942
;26943
;26944
;26945
;26946
;26947
;26948
;26949
;26950
;26951
;26952
;26953
;26954
;26955
;26956
;26957
;26958
;26959
;26960
;26961
;26962
;26963
;26964
;26965
;26966
;26967
;26968
;26969
;26970
;26971
;26972
;26973
;26974
;26975
;26976
;26977
;26978
;26979
;26980
;26981
;26982
;26983
;26984
;26985
;26986
;26987
;26988
;26989
;26990
;26991
;26992
```

Resources (Above, plus:)

GPR2	MTEMP9 in byte#0
TEMP0 & TEMP1	Partial Product result from multiplication of a source 1 longword and a source 2 longword, MSD in byte#0 of TEMP1, LSB in byte#3 of TEMP0
TEMP2	Misc. data
TEMP3-TEMP6	Holds Source 2, memory format, LSB in byte#3 of TEMP3, MSB in byte#0 of TEMP6
TEMP7	Binary value of src.#2 longword
RTEMP8	Destination Length
RTEMP9	Destination Longword counter, values from 0-3 indicate which destination longword we are adding to:

0	1	2	3

and values -1 to -3 indicate we are pointing to "imaginary" longwords where ov. can occur

RTEMP10-RTEMP13	Holds Source #1, memory format, see description of source #2
MM.TEMP0	Misc. data
MM.TEMP1	Points to current source 1 longword as a 0-3 code, see description of destination pointer
MM.TEMP2	Misc. data
MM.TEMP5	Ditto *MM.TEMP0 is not needed after any call to IE.SERV.IP.TS2. MM.TEMPx are not referenced after any BUS exception or an int. is found pending*
MTEMP8-MERRCOD	Destination data, memory format, LSB in byte#3 of MTEMP8, MSB in byte#0 of MERRCOD. *MERRCOD is not referenced after any BUS exception.*
FPDOFFSET	ZLIT0[16.]
STEPC	I/O routine counter
RNUM	Points to current Source #2 TEMP
DREG	Copy of RNUM
QREG	Misc. data in Source 2-to-binary routine and DS.WRITF
FLAG0	DS.WRITE temp., masks PSL<C>
FLAG1	<V>image
FLAG2	1) Source 1 sign 2) <Z>image
FLAG3	1) Source 2 sign 2) <N>image

```
                    ;26993          ;       Output          GPR0            0
                    ;26994          ;                       GPR1            Base address, source 1
                    ;26995          ;                       GPR2            0
                    ;26996          ;                       GPR3            Base address, source 2
                    ;26997          ;                       GPR4            0
                    ;26998          ;                       GPR5            Base address, destination
                    ;26999          ;                       CC              C_0, NZV set/cleared
                    ;27000          ;
                    ;27001          ;*******************************************************************************
```

```
                                ;27002 .TOC "  Decimal String                    :       Initialization after ADDP4/SUBP4"
                                ;27003
                                ;27004 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;27005 DS.MULP:
                                ;27006          ;-------------------------------;
U 1183, 0C84,9592,4B74,9847,010F,9 ;27007       R[R2]_M[TEMP9],ALUS?            ;Save Source #2 length for possible
                                ;27008                                          ;pack-up, what is source 1 sign?
                                ;27009
                                ;27010 =01
                                ;27011          ;01-----------------------------;
                                ;27012          CLEAR FLAG2,                    ;Positive, begin to zero dest. temps
                                ;27013          M[TEMP8]_ZLIT0[0],
U 10F9, 0916,8037,0030,0047,0105,4 ;27014       NEXT/DS.MULP.2
                                ;27015
                                ;27016          ;11-----------------------------;
U 10FB, 0156,8037,0030,0047,0105,4 ;27017       SET FLAG2,M[TEMP8]_ZLIT0[0]     ;Negative, ditto
                                ;27018
                                ;27019 =0*
                                ;27020 DS.MULP.2:
                                ;27021          ;0*-----------------------------;
                                ;27022          PUSH,M[TEMP7]_ZLIT0[0],         ;Divide Src.#1 by 10. to slide
U 1054, 0186,8037,0030,0047,052B,8 ;27023       NEXT/DS.DIV10                   ;sign nibble away
                                ;27024
                                ;27025          ;1*-----------------------------;
U 1056, 0986,0037,0030,8047,0118,4 ;27026       M[FPDOFFSET]_ZLIT0[16.]         ;New MULP IANDE offset
                                ;27027
                                ;27028          ;-------------------------------;
U 1184, 0C84,3592,4032,8047,0118,6 ;27029       R[TEMP10]_M[TEMP3]              ;Save Source 1
                                ;27030
                                ;27031          ;-------------------------------;
U 1186, 0484,4592,4032,C047,0118,8 ;27032       R[TEMP11]_M[TEMP4]
                                ;27033
                                ;27034          ;-------------------------------;
U 1188, 0C84,5592,4033,0047,0118,A ;27035       R[TEMP12]_M[TEMP5]
                                ;27036
                                ;27037          ;-------------------------------;
                                ;27038          R[TEMP13]_M[TEMP6],             ;Now read source 2
U 118A, 0884,6592,4033,4047,0113,D ;27039       NEXT/DS.MULP.READ
                                ;27040
                                ;27041 =00
                                ;27042          ;00-----------------------------;
                                ;27043          M[TEMP9]_ZLIT0[0],PUSH,         ;Zero a dest. TEMP, divide source 2
U 113C, 0586,9037,0030,0047,0528,8 ;27044       NEXT/DS.DIV10                   ;by 10 to slide sign nibble out
                                ;27045
                                ;27046 DS.MULP.READ:
                                ;27047          ;01-----------------------------;
U 113D, 0880,C0A1,0034,C4A7,0505,D ;27048       PUSH,VA_D+R[R3]+1,NEXT/DS.READ  ;Read Source 2, RETURN-1
                                ;27049
                                ;27050          ;10-----------------------------;
U 113F, 0586,A037,0030,0047,0118,F ;27051       M[TEMP10]_ZLIT0[0]
                                ;27052 =
                                ;27053
                                ;27054          ;-------------------------------;
U 118F, 048C,D3B7,0B70,1847,0110,5 ;27055       M[ERRCOD]_0,ALUS?               ;All dest. temps._0, what was
                                ;27056                                          ;source 2 sign?
```

```
                                ;27057 .TOC     "       Decimal String                  :       Multiply loops"
                                ;27058
                                ;27059 =01
                                ;27060          ;01-------------------------------;
                                ;27061          CLEAR FLAG3,                    ;Positive, RNUM_D_points to low
                                ;27062          RNUM_D_ZLIT0[3],                ;LONGWORD of source 2
U 1105, 0519,0C35,2030,1847,0119,3      ;27063          NEXT7DS.MULP.TOP
                                ;27064
                                ;27065          ;11-------------------------------;
                                ;27066          SET FLAG3,                      ;Negative, ditto
U 1107, 0D59,0C35,2030,1847,0119,3      ;27067          RNUM_D_ZLIT0[3]
                                ;27068
                                ;27069 DS.MULP.TOP:
                                ;27070          ;-------------------------------;
U 1195, 0884,0582,4033,C047,0119,8      ;27071          R[MM.TEMP1]_D                   ;Source 1 TEMP counter_3 to point to
                                ;27072                                          ;first source 1 longword
                                ;27073
                                ;27074          ;-------------------------------;
U 1198, 0484,0582,4B32,58E7,0121,4      ;27075          R[TEMP9]_D,IP.TS?               ;Init. dest. longword counter to 3
                                ;27076
                                ;27077
                                ;27078          ;*****************************************************************************
                                ;27079          ; Top of outer multiply loop.  RNUM points to a legitimate source 2
                                ;27080          ; longword, if it's zero we skip over this pass thru the loop.  Here
                                ;27081          ; on IP.TS? split.
                                ;27082          ;*****************************************************************************
                                ;27083 =0
                                ;27084 DS.MULP.LOP2:
                                ;27085          ;0-------------------------------;
                                ;27086          R[TEMP2]_M[TEMP.R].BCDSWP,      ;Get arithmetic format of next/first
U 1214, 0C85,0637,0A70,8047,0114,0      ;27087          WX.NE.0?,NEXT/DS.MULP.LOP2.2    ;word, is it zero?
                                ;27088
                                ;27089          ;1-------------------------------;
                                ;27090          PUSH,INTPEND OR TIMER?,         ;Something pending, return -1 or
U 1215, 0480,0036,4AF0,0047,04F7,0      ;27091          NEXT/IE.SERV.IP.TS2             ;branch to IE.PACK.DONE
                                ;27092
                                ;27093
                                ;27094 =00
                                ;27095 DS.MULP.LOP2.2:
                                ;27096          ;00-------------------------------;
                                ;27097          R[TEMP9]_R8-1,WB<31-30>?,       ;Yes, skip this loop pass, dec.
U 1140, 0084,0E7D,06F2,4047,0915,1 371* ;27098          NEXT/DS.MULP.END                ;dest. counter, are we done?
                                ;27099
                                ;27100          ;01-------------------------------;
                                ;27101          PUSH,M[MM.TEMP0]_ZLIT0[0F],     ;No, load nibble mask, TEMP7_
U 1141, 0C86,0C37,0030,7847,051E,2      ;27102          NEXT/DS.MULP.SRC2BIN            ;binary value of TEMP2
```

```
                                ;27103          ;*************************************************************************
                                ;27104          ; Top of Inner (Source 1) loop.  The beginning TEMP for adding
                                ;27105          ; to the dest. parallels which TEMP we are reading from source 2, but
                                ;27106          ; not source 1..after splitting to decide which portion to multiply,
                                ;27107          ; we split again to decide where, if anywhere, to add it.
                                ;27108          ;*************************************************************************
                                ;27109
                                ;27110 DS.MULP.LOP1:
                                ;27111          ;10-----------------------------;
U 1142, 0480,05BE,4273,C047,0114,4  ;27112      WB_R[MM.TEMP1],WB<1-0>?         ;Which source 1 TEMP are we?
                                ;27113 =
                                ;27114
                                ;27115 =00
                                ;27116          ;00-----------------------------;
                                ;27117          M[TEMP0]_R[TEMP13],             ;Last Source 1 longword, is it zero?
U 1144, 0486,05BE,4A73,4047,0114,8  ;27118      NEXT/DS.MULP.LOP1.MUL,WX.NE.0?
                                ;27119
                                ;27120          ;01-----------------------------;
                                ;27121          M[TEMP0]_R[TEMP12],             ;Second-to-last source 1 longword,
U 1145, 0086,05BE,4A73,0047,0114,8  ;27122      NEXT/DS.MULP.LOP1.MUL,WX.NE.0?  ;is it zero?
                                ;27123
                                ;27124          ;10-----------------------------;
                                ;27125          M[TEMP0]_R[TEMP11],             ;Second source 1 longword,
U 1146, 0486,05BE,4A72,C047,0114,8  ;27126      NEXT/DS.MULP.LOP1.MUL,WX.NE.0?  ;is it zero?
                                ;27127
                                ;27128          ;11-----------------------------;
                                ;27129          M[TEMP0]_R[TEMP10],             ;First source 1 longword,
U 1147, 0086,05BE,4A72,8047,0114,8  ;27130      WX.NE.0?                        ;is it zero?
                                ;27131
                                ;27132
                                ;27133 =00
                                ;27134 DS.MULP.LOP1.MUL:
                                ;27135          ;00-----------------------------;
                                ;27136          R[MM.TEMP1]_RB-1,WB<31-30>?,    ;Zero Source 1 temp., don't multiply or
U 1148, 0084,0E7D,C6F3,C047,0910,D 371*  ;27137 NEXT/DS.MULP.LOP1.END            ;add it, is it the last source 1 TEMP?
                                ;27138
                                ;27139          ;01-----------------------------;
                                ;27140          R[TEMP1]_0,PUSH,                ;Nonzero, high partial prod._0,
U 1149, 0084,05B7,0030,4047,051D,3  ;27141      NEXT/DS.MULP.MUL                ;multiply it by source 2 temp value in
                                ;27142                                          ;TEMP7
                                ;27143
                                ;27144          ;10-----------------------------;
U 114A, 0080,05BE,4232,4047,0103,8  ;27145      WB_R[TEMP9],WB<5-0>?            ;Into which dest. temp(s). does this
                                ;27146                                          ;get added?
                                ;27147 =
```

```
                                ;27148          ;*****************************************************************
                                ;27149          ; Sources have been multiplied, partial product in TEMP0 and TEMP1.
                                ;27150          ; Here from WB<5-0>? BUT on RTEMP9, which indicates which destination
                                ;27151          ; TEMP we begin adding to during a loop pass.
                                ;27152          ; OV can occur here in 2 ways:
                                ;27153          ; 1) Carryout from adding partial product to dest. temps.
                                ;27154          ; 2) Dest. pointer (RTEMP9) <0 (-1, -2, -3).  The current source 2
                                ;27155          ;    temp. and source 1 temp. are nonzero, hence if we are pointing
                                ;27156          ;    past the highest dest. temp. we know we have ov.
                                ;27157          ;*****************************************************************
                                ;27158 =111000
                                ;27159          ;111000-------------------------;
                                ;27160          M[ERRCOD] (MB+R[TEMP0]).BCD,    ;(0)Last Dest. longword
U 1038, 0086,B00?,4020,0047,0119,A ;27161       NEXT/DS.MULP.LOP1.0
                                ;27162
                                ;27163          ;111001-------------------------;
                                ;27164          M[TEMP10] (MB+R[TEMP0]).BCD,    ;(1)Second-to-Last dest. longword
U 1039, 0486,A001,4020,0047,0119,F ;27165       NEXT/DS.MULP.LOP1.1
                                ;27166
                                ;27167          ;111010-------------------------;
                                ;27168          M[TEMP9] (MB+R[TEMP0]).BCD,     ;(2)Second dest. longword
U 103A, 0486,9001,4020,0047,011A,C ;27169       NEXT/DS.MULP.LOP1.2
                                ;27170
                                ;27171          ;111011-------------------------;
                                ;27172          M[TEMP8] (MB+R[TEMP0]).BCD,     ;(3)First dest. longword
U 103B, 0886,8001,4020,0047,011A,E ;27173       NEXT/DS.MULP.LOP1.3
                                ;27174
                                ;27175 =111101
                                ;27176          ;111101-------------------------;
                                ;27177          SET FLAG1,R[MM.TEMP1]_RB-1,     ;(-3)We are pointing past the dest.
                                ;27178          WB<31-30>?,                     ;temps., *OV*, dec. counter & is loop
U 103D, 084C,0E7D,06F3,C047,0910,D 371* ;27179  NEXT/DS.MULP.LOP1.END             ;finished?
                                ;27180
                                ;27181          ;111110-------------------------;
                                ;27182          SET FLAG1,R[MM.TEMP1]_RB-1,     ;(-2), ditto
U 103E, 084C,0E7D,06F3,C047,0910,D 371* ;27183  WB<31-30>?,NEXT/DS.MULP.LOP1.END
                                ;27184
                                ;27185          ;111111-------------------------;
                                ;27186          SET FLAG1,R[MM.TEMP1]_RB-1,     ;(-1), ditto
U 103F, 084C,0E7D,06F3,C047,0910,D 371* ;27187  WB<31-30>?,NEXT/DS.MULP.LOP1.END
```

```
                                        ;27188          ;*************************************************************************
                                        ;27189          ; RTEMP9=0, TEMP0 was added to MERRCOD.  Ov occurs if TEMP1+ALKC>0 OR
                                        ;27190          ; CARRYOUT after adding to TEMP1.
                                        ;27191          ;*************************************************************************
                                        ;27192
                                        ;27193 DS.MULP.LOP1.0:
                                        ;27194          ;-------------------------------;
                                        ;27195          WB (M[TEMP1]+ALKC).BCD,
                                        ;27196          ALUS_UNSGN,WX.NE.0?,
U 119A, 0C80,1041,4A6D,90C7,0914,C 364* ;27197          NEXT/DS.MULP.LOP1.ALUS
                                        ;27198
                                        ;27199          ;*************************************************************************
                                        ;27200          ; RTEMP9=1, TEMP0 was added to MTEMP10.  Add TEMP1 to MERRCOD, store
                                        ;27201          ; CARRYOUT data in ALUS.
                                        ;27202          ;*************************************************************************
                                        ;27203
                                        ;27204 DS.MULP.LOP1.1:
                                        ;27205          ;-------------------------------;
                                        ;27206          M[ERRCOD] (MB+R[TEMP1]+ALKC).BCD,
U 119F, 0486,B041,4020,50C7,0114,C      ;27207          ALUS_UNSGN,NEXT/DS.MULP.LOP1.ALUS
                                        ;27208
                                        ;27209          ;*************************************************************************
                                        ;27210          ; RTEMP9=2, TEMP0 was added to MTEMP9.  Add TEMP1 to MTEMP10, CARRYOUT
                                        ;27211          ; to MERRCOD, store CARRYOUT data in ALUS.
                                        ;27212          ;*************************************************************************
                                        ;27213
                                        ;27214 DS.MULP.LOP1.2:
                                        ;27215          ;-------------------------------;
                                        ;27216          M[TEMP10] (MB+R[TEMP1]+ALKC).BCD,
U 11AC, 0486,A041,4020,4047,011B,2      ;27217          NEXT/DS.MULP.LOP1.23
                                        ;27218
                                        ;27219          ;*************************************************************************
                                        ;27220          ; RTEMP9=3, TEMP0 was added to MTEMP8.  Add TEMP1 to MTEMP9, carry
                                        ;27221          ; to higher TEMPs, store CARRYOUT data in ALUS.
                                        ;27222          ;*************************************************************************
                                        ;27223
                                        ;27224 DS.MULP.LOP1.3:
                                        ;27225          ;-------------------------------;
U 11AE, 0C86,9041,4020,4047,011B,0      ;27226          M[TEMP9]_(MB+R[TEMP1]+ALKC).BCD
                                        ;27227
                                        ;27228          ;-------------------------------;
U 11B0, 0086,A041,402D,8047,011B,2      ;27229          M[TEMP10]_(MB+ALKC).BCD
                                        ;27230
                                        ;27231 DS.MULP.LOP1.23:
                                        ;27232          ;-------------------------------;
                                        ;27233          M[ERRCOD]_(MB+ALKC).BCD,        ;RTEMP9 codes 2 and 3 join here
U 11B2, 0086,B041,402D,90C7,0114,C      ;27234          ALUS_UNSGN
```

```
                                ;27235          ;*****************************************************************************
                                ;27236          ; End of Source 1 & Source 2 loops.  The parital product in TEMP0 and
                                ;27237          ; TEMP1 has been added to the appropriate destination TEMPs or checked
                                ;27238          ; for ov.  Enter at DS.MULP.LOP1.ALUS:
                                ;27239          ; 1) Directly with CARRYOUT data in ALUS.
                                ;27240          ; 2) On a WX.NE.0? BUT with CARRYOUT data in ALUS from adding TEMP0
                                ;27241          ;    to M[ERRCOD], the highest destination TEMP.
                                ;27242          ;
                                ;27243          ; Enter at DS.MULP.LOP1.END:
                                ;27244          ; 1) From =01 and =11 splits in DS.MULP.LOP1.ALUS: code on a WB<31-30>?
                                ;27245          ;    BUT.  =01 is back down Source 1 by a longword, =11 is we have just
                                ;27246          ;    finished a pass thru source 1, consider backing down Source 2.
                                ;27247          ; 2) From =111101, =111110, and =111111 splits in DS.MULP.LOP1.MUL(+3)
                                ;27248          ;    code on a WB<31-30>? BUT.  Same splits as reason #1, having come
                                ;27249          ;    from an overflow-from-extra-longwords case.
                                ;27250          ;*****************************************************************************
                                ;27251 =00
                                ;27252 DS.MULP.LOP1.ALUS:
                                ;27253                  ;00-----------------------------;
U 114C, 0480,0036,4B70,1847,0114,D ;27254               ALUS?                           ;Was there OV?
                                ;27255
                                ;27256                  ;01-----------------------------;
                                ;27257                  WB<31-30>?,SET FLAG1,           ;OV from CARRYOUT (ALUS BUT) or
                                ;27258                  R[MM.TEMP1]_RB-1,               ;TEMP1>0 from RTEMP9=0 code,
U 114D, 084C,0E7D,06F3,C047,0910,D 371* ;27259          NEXT/DS.MULP.LOP1.END           ;<V>image_1, dec. source 1 counter,
                                ;27260                                                  ;have we finished a pass through all of
                                ;27261                                                  ;source 1?
                                ;27262 =11
                                ;27263                  ;11-----------------------------;
U 114F, 0084,0E7D,06F3,C047,0910,D 371* ;27264          R[MM.TEMP1]_RB-1,WB<31-30>?     ;No OV from CARRYOUT, dec. counter and
                                ;27265                                                  ;check it
                                ;27266
                                ;27267
                                ;27268 =01
                                ;27269 DS.MULP.LOP1.END:
                                ;27270                  ;01-----------------------------;
                                ;27271                  R[TEMP9]_RB-1,                  ;Not yet finished with "inside loop",
U 110D, 0084,0E7D,0032,4047,0114,2 ;27272               NEXT/DS.MULP.LOP1               ;back down dest., loop back
                                ;27273
                                ;27274                  ;11-----------------------------;
U 110F, 0084,073D,06D2,4047,0915,1 376* ;27275          R[TEMP9]_RB+CONX(2),WB<31-30>?  ;Finished pass thru source 1, inc.
                                ;27276                                                  ;dest. pointer, check it
```

```
                                ;27277 .TOC     "       Decimal String                  :       End of loops, Destination wr
                                ;27278
                                ;27279          ;*****************************************************************************
                                ;27280          ; RTEMP9 was 0 - -3.  Adding 2 gives 2 - -1, which indicates which TEMP
                                ;27281          ; we would add to next pass thru the outer (source 2) loop.  If RTEMP9
                                ;27282          ; = -1, we are finished. We also BUT to here from skipping over a
                                ;27283          ; zero Source2 TEMP.
                                ;27284          ;*****************************************************************************
                                ;27285 =01
                                ;27286 DS.MULP.END:
                                ;27287          ;01-------------------------------;
                                ;27288          RNUM_D_D+1,                     ;>=0, ok, inc. source 2 pointer to
U 1151, 0081,D0A1,203D,8047,011C,0      ;27289          NEXT7DS.MULP.LOP2.LOOP          ;next source 2 TEMP
                                ;27290
                                ;27291          ;11-------------------------------;
                                ;27292          R[TEMP3]_M[TEMP8],IP.TS?,       ;<0, **All done**, begin moving
U 1153, 0C84,8592,4B30,D8E7,0121,6      ;27293          NEXT/DS.MULP.FINISH             ;destination to DS.WRITE TEMPs
                                ;27294
                                ;27295
                                ;27296 DS.MULP.LOP2.LOOP:
                                ;27297          ;-------------------------------;
U 11C0, 0884,0737,0013,C047,011C,2      ;27298          R[MM.TEMP1]_CONX(2)             ;MM.TEMP1_3 to point to beginning of
                                ;27299                                          ;source 1, loop back & split if
                                ;27300                                          ;anything pending
                                ;27301
                                ;27302          ;-------------------------------;
                                ;27303          R[MM.TEMP1]_RB+CONX(1),
U 11C2, 0084,073D,0B03,D8E7,0121,4      ;27304          NEXT/DS.MULP.LOP2,IP.TS?
                                ;27305
                                ;27306
                                ;27307
                                ;27308          ;*****************************************************************************
                                ;27309          ; =0 is nothing pending or timer service, =1 is something pending,
                                ;27310          ; RETURN -1 or branch to IE.PACK.DONE
                                ;27311          ;*****************************************************************************
                                ;27312 =0
                                ;27313 DS.MULP.FINISH:
                                ;27314          ;0-------------------------------;
                                ;27315          R[TEMP4]_M[TEMP9],
U 1216, 0084,9592,4031,0047,011C,A      ;27316          NEXT/DS.MULP.FINISH.1
                                ;27317
                                ;27318          ;1-------------------------------;
                                ;27319          PUSH,INTPEND OR TIMER?,
U 1217, 0480,0036,4AF0,0047,04F7,0      ;27320          NEXT/IE.SERV.IP.TS2
                                ;27321
                                ;27322
                                ;27323 DS.MULP.FINISH.1:
                                ;27324          ;-------------------------------;
U 11CA, 0484,A592,4031,4047,011C,F      ;27325          R[TEMP5]_M[TEMP10]
                                ;27326
                                ;27327          ;-------------------------------;
U 11CF, 0884,B592,4031,8047,0115,4      ;27328          R[TEMP6]_M[ERRCOD]
```

```
                        ;27329 =00
                        ;27330          ;00--------------------------------;
                        ;27331          WB_M[ERRCOD].AND.ZLIT0[0F0],    ;PUSH for DS.MUL10 RETURN+2, is high
                        ;27332          WX.NE.0?,PUSH,                  ;nibble zero?
U 1154, 0180,BC12,0A77,8047,0521,8      ;27333          NEXT/DS.MULP.FINISH.2
                        ;27334
                        ;27335          ;01--------------------------------;
                        ;27336          D_R[TEMP8].SR.1,FLAG<2-0>?,     ;Dest. is written, set up DREG for -0
U 1155, 0080,05BD,A5B2,004' ..05,8      ;27337          NEXT/DS.ADDP4.FLAGS             ;conversion, split on CC image to
                        ;27338                                          ;ADDP4 code
                        ;27339
                        ;27340          ;10--------------------------------;
                        ;27341          VA_R[R5].PUSH,                  ;Load dest. base address, split on
                        ;27342          FLAG1 (FLAG2.XOR.FLAG3)?,       ;old FLAG values of source 1 & 2 signs,
                        ;27343          SET FLAG2,                      ;PUSH for DS.WRITE RETURN [-1]
U 1156, 0C50,05BE,4575,44A7,0500,A      ;27344          NEXT/DS.MULP.FINISH.SIGN
                        ;27345 =
                        ;27346
                        ;27347
                        ;27348          ;*****************************************************************************
                        ;27349          ; =0 is high nibble of dest. is zero, no ov.
                        ;27350          ; =1 is high nibble of dest. is nonzero, ov.
                        ;27351          ; STEPC_Q_DS.WRITE data & we branch to DS.MUL10 to clear sign nibble.
                        ;27352          ;*****************************************************************************
                        ;27353 =0
                        ;27354 DS.MULP.FINISH.2:
                        ;27355          ;0--------------------------------;
                        ;27356          STEPC_Q_R[TEMP8].SR.1,
U 1218, 0880,0E7E,9032,0107,012B,C      ;27357          NEXT/DS.MUL10
                        ;27358
                        ;27359          ;1--------------------------------;
                        ;27360          STEPC_Q_R[TEMP8].SR.1,          ;<V>image_1
U 1219, 0048,0E7E,9032,0107,012B,C      ;27361          NEXT/DS.MUL10,SET FLAG1
                        ;27362
                        ;27363
                        ;27364          ;*****************************************************************************
                        ;27365          ; Here on XOR BUT of FLAG3 and old FLAG2 values.
                        ;27366          ; =10 is dest. is positive, mask pos. sign & <N>image_0.
                        ;27367          ; =11 is dest. is negative, mask neg. sign & <N>image_1.
                        ;27368          ; Note that the PUSH for DS.WRITE was made at DS.MULP.FINISH.1+4
                        ;27369          ; and that the -0 dest. check is made in ADDP4/SUBP4 code.
                        ;27370          ;*****************************************************************************
                        ;27371 =10
                        ;27372 DS.MULP.FINISH.SIGN:
                        ;27373          ;10--------------------------------;
                        ;27374          M[TEMP3]_MB.OR.ZLIT24[0C],
U 100A, 051E,3C92,4030,6047,012C,E      ;27375          CLEAR FLAG3,NEXT/DS.WRITE
                        ;27376
                        ;27377          ;11--------------------------------;
                        ;27378          M[TEMP3]_MB.OR.ZLIT24[0D],
U 100B, 095E,3C92,4030,6847,012C,E      ;27379          SET FLAG3,NEXT/DS.WRITE
```

```
                        ;27380 .TOC "  Decimal String                    :       Multiplication routine"
                        ;27381
                        ;27382          ;*****************************************************************************
                        ;27383          ;
                        ;27384          ; Double-and-Divide is used to multiply TEMP7(binary) by TEMP0&TEMP1
                        ;27385          ; (Packed Decimal).
                        ;27386          ;
                        ;27387          ;       Input           TEMP0           Packed decimal data, memory
                        ;27388          ;                                       format
                        ;27389          ;                       TEMP1           Zero (0)
                        ;27390          ;                       TEMP7           Binary multiplier
                        ;27391          ;                       FPDOFFSET       Nonzero
                        ;27392          ;                       Int. check      Made no more than <max. loop -
                        ;27393          ;                                       7> cycles ago
                        ;27394          ;
                        ;27395          ;       Resources       TEMP2           Copy of TEMP7
                        ;27396          ;                       MM.TEMP2,       Accumulates product, memory
                        ;27397          ;                       MM.TEMP5        format, MSD in MM.TEMP2
                        ;27398          ;
                        ;27399          ;       Output          TEMP0-TEMP1     Product, MSD in TEMP1, LSD in
                        ;27400          ;                                       TEMP0, memory format
                        ;27401          ;                       TEMP2           Zero (0)
                        ;27402          ;                       TEMP7           Untouched
                        ;27403          ;                       Int. check      Made during multiplication.
                        ;27404          ;                                       Microword on RETURN is cycle
                        ;27405          ;                                       #3.
                        ;27406          ;
                        ;27407          ;       Returns         +1
                        ;27408          ;
                        ;27409          ; TEMP7 is assumed to be a binary value <=99999999.  If TEMP1 is
                        ;27410          ; zero, and TEMP0 has valid Packed Decimal data, no ov can occur
                        ;27411          ; since 99999999. x 99999999. < 9999999900000000.
                        ;27412          ;
                        ;27413          ;*****************************************************************************
                        ;27414
                        ;27415 DS.MULP.MUL:
                        ;27416          ;--------------------------------;
U 11D3, 0084,05B7,0033,8047,011D,7      ;27417          R[MM.TEMP5]_0                   ;Init. product accumulators
                        ;27418
                        ;27419          ;--------------------------------;
U 11D7, 0084,05B7,0039,8047,011D,8      ;27420          R[MM.TEMP2]_0
                        ;27421
                        ;27422          ;--------------------------------;
U 11DB, 0C86,25BE,42B1,C047,0121,A      ;27423          M[TEMP2]_R[TEMP7],WB<0>'        ;Is multiplier odd?
```

```
                                        ;27424 =0
                                        ;27425 DS.MULP.MUL.LOP:
                                        ;27426          ;0--------------------------------;
                                        ;27427          WB_M[TEMP2],WX.NE.0?,           ;No, could be even or zero
U 121A, 0880,2592,4A70,0047,0121,C      ;27428          NEXT/DS.MULP.MUL.EZER
                                        ;27429
                                        ;27430          ;1--------------------------------;
U 121B, 0C84,0001,4023,8047,011D,F      ;27431          R[MM.TEMP5]_(M[TEMP0]+RB).BCD   ;Yes!, Partial product_par. prod.
                                        ;27432                                          ;plus CURRENT multiplicand
                                        ;27433
                                        ;27434          ;--------------------------------;
                                        ;27435          R[MM.TEMP2]_(M[TEMP1]+RB+ALKC).BCD,
U 11DF, 0484,1041,4029,8047,0121,D      ;27436          NEXT/DS.MULP.MUL.DD
                                        ;27437
                                        ;27438 =0
                                        ;27439 DS.MULP.MUL.EZER:
                                        ;27440          ;0--------------------------------;
                                        ;27441          M[TEMP0]_R[MM.TEMP5],IP.TS?,    ;Multiplier is zero, TEMP0 & TEMP1
U 121C, 0C86,05BE,4B33,98E7,0122,0      ;27442          NEXT/DS.MULP.MUL.EXIT           ;partial product, anything pending?
                                        ;27443
                                        ;27444 DS.MULP.MUL.DD:
                                        ;27445          ;1--------------------------------;
U 121D, 0C86,0001,4020,0047,011E,0      ;27446          M[TEMP0]_(MB+R[TEMP0]).BCD      ;Double multiplicand
                                        ;27447
                                        ;27448          ;--------------------------------;
                                        ;27449          M[TEMP1]_(MB+R[TEMP1]+ALKC).BCD,;Anything pending?
U 11E0, 0486,1041,4B20,56E7,0121,E      ;27450          IP.TS?
                                        ;27451
                                        ;27452 =0
                                        ;27453          ;0--------------------------------;
                                        ;27454          M[TEMP2]_MB.SR.1,WB<0>?,         ;No, divide multiplier by 2, is it odd?
U 121E, 0486,2001,82BD,8047,0921,A 355* ;27455          NEXT/DS.MULP.MUL.LOP
                                        ;27456
                                        ;27457          ;1--------------------------------;
                                        ;27458          PUSH,INTPEND OR TIMER?,         ;Yes, RETURN-1 or to IE.PACK.DONE
U 121F, 0480,0036,4AF0,0047,04F7,0      ;27459          NEXT/IE.SERV.IP.TS2
                                        ;27460
                                        ;27461
                                        ;27462 =0
                                        ;27463 DS.MULP.MUL.EXIT:
                                        ;27464          ;0--------------------------------;
                                        ;27465          M[TEMP1]_R[MM.TEMP2],           ;Nothing pending or timer service,
U 1220, 0886,15BE,40B9,8047,0000,1      ;27466          RETURN [T]                      ;RETURN+1
                                        ;27467
                                        ;27468          ;1--------------------------------;
                                        ;27469          PUSH,NEXT/IE.SERV.IP.TS2,       ;Something pending, return -1 or
U 1221, 0480,0036,4AF0,0047,04F7,0      ;27470          INTPEND OR TIMER?               ;never..
```

```
                                ;27471 .TOC "  Decimal String                    :       Source#2-to-binary routine"
                                ;27472
                                ;27473          ;*************************************************************************
                                ;27474          ;
                                ;27475          ; DS.MULP.SRC2BIN converts an 8-digit Packed Decimal value into binary.
                                ;27476          ; No ov. can occur since 99999999 < 2^27.
                                ;27477          ;
                                ;27478          ;       Input           MM.TEMP0        0000000F
                                ;27479          ;                       TEMP2           PD data, BCDSWP'd into
                                ;27480          ;                                       arithmetic format
                                ;27481          ;                                       [MDR:=ABCDEFGH]
                                ;27482          ;
                                ;27483          ;       Resources       QREG            Holding digits
                                ;27484          ;                       MM.TEMP2        Constructing value
                                ;27485          ;
                                ;27486          ;       Output          TEMP7           Binary value of TEMP2
                                ;27487          ;
                                ;27488          ;       Return          +1
                                ;27489          ;
                                ;27490          ;*************************************************************************
                                ;27491
                                ;27492 DS.MULP.SRC2BIN:
                                ;27493          ;-------------------------------;
U 11E2, 0804,2387,0021,C047,0115,8 ;27494       R[TEMP7]_M[TEMP2].RR.24         ;Nibble#1_"A"
                                ;27495
                                ;27496 =00
                                ;27497          ;00-------------------------------;
                                ;27498          R[TEMP7] ((M[MM.TEMP0] RB).RR.4.AND.MB).SL.1,   ;TEMP7_"A"'0,
U 1158, 0484,D252,C031,C047,053D,E ;27499       PUSH,NEXT/DS.SR.BIN.MUL10..2                    ;finish multiply by 10
                                ;27500
                                ;27501          ;01-------------------------------;
                                ;27502          Q_M[MM.TEMP0].AND.(R[TEMP2].RR.24),     ;Q_"B",
U 1159, 0080,D292,1020,8047,053D,D ;27503       PUSH,NEXT/DS.SR.BIN.MUL10.              ;add and multiply by 10
                                ;27504
                                ;27505          ;10-------------------------------;
U 115A, 0084,23B7,0019,8047,0115,C ;27506       R[MM.TEMP2]_M[TEMP2].RR.16      ;TEMP7 has AB0, MM.TEMP2_"C" in
                                ;27507                                          ;nibble#1
                                ;27508 =
                                ;27509
                                ;27510 =00
                                ;27511          ;00-------------------------------;
                                ;27512          Q_M[MM.TEMP0].AND.((MB R[MM.TEMP2]).RR.4),      ;Q_"C", add and
U 115C, 0480,D252,1039,8047,053D,D ;27513       PUSH,NEXT/DS.SR.BIN.MUL10.                      ;multiply by 10
                                ;27514
                                ;27515          ;01-------------------------------;
                                ;27516          Q_M[MM.TEMP0].AND.R[MM.TEMP2],  ;Q_"D", add and multiply by 10.
U 115D, 0080,D002,1039,8047,053D,D ;27517       PUSH,NEXT/DS.SR.BIN.MUL10.
                                ;27518
                                ;27519          ;10-------------------------------;
U 115E, 0484,23B7,0009,8047,0116,0 ;27520       R[MM.TEMP2]_M[TEMP2].RR.8       ;TEMP7 has ABCD0, MM.TEMP2_"E" in
                                ;27521                                          ;nibble#1
                                ;27522 =
```

```
                                        ;27523 =00
                                        ;27524          ;00--------------------------------;
                                        ;27525          Q_M[MM.TEMP0].AND.((MB R[MM.TEMP2]).RR.4),      ;Q_"E", add and multiply
U 1160, 0480,D252,1039,8047,053D,D      ;27526          PUSH,NEXT/DS.SR.BIN.MUL10.                      ;by 10.
                                        ;27527
                                        ;27528          ;01--------------------------------;
                                        ;27529          Q_M[MM.TEMP0].AND.R[MM.TEMP2],  ;Q_"F", add and multiply by 10.
U 1161, 0080,D002,1039,8047,053D,D      ;27530          PUSH,NEXT/DS.SR.BIN.MUL10.
                                        ;27531
                                        ;27532          ;10--------------------------------;
                                        ;27533          Q_M[MM.TEMP0].AND.((MB R[TEMP2]).RR.4), ;Q_"G", add and multiply by 10.
U 1162, 0480,D252,1030,8047,053D,D      ;27534          PUSH,NEXT/DS.SR.BIN.MUL10.
                                        ;27535
                                        ;27536          ;11--------------------------------;
U 1163, 0080,D002,1030,8047,011E,4      ;27537          Q_M[MM.TEMP0].AND.R[TEMP2]      ;TEMP7 has ABCDEFG0, Q_"H"
                                        ;27538
                                        ;27539          ;--------------------------------;
U 11E4, 0C84,0039,00B1,C047,U000,1      ;27540          R[TEMP7]_RB+Q,RETURN [1]        ;TEMP7 has ABCDEFGH, RETURN
```

```
;27541 .TOC "  Decimal String                    : DIVP"
;27542
;27543          ;*****************************************************************************
;27544          ;
;27545          ;       DIVP divrlen.rw, divraddr.ab, divdlen.rw, divdaddr.ab,
;27546          ;               divdaddr.ab, quolen.rw, quoaddr.ab
;27547          ;
;27548          ;       Instruction performs operand fetches and preliminary
;27549          ;       initialization in <ADD-SUB>P6.
;27550          ;
;27551          ;       Done via restoring divide.  The following notations
;27552          ;       used throughout the instruction:
;27553          ;
;27554          ;       Q=DV/DR :       What we're doing
;27555          ;       Q(LB)   :       Length of Q, bytes
;27556          ;       Q(LD)   :       Length of Q, digits, no leading zeros
;27557          ;       DV(LB)  :       Length of DV, bytes
;27558          ;       DV(LD)  :       Length of DV, digits
;27559          ;       DV(LDNLZ):      Length of DV, digits, no leading zeros
;27560          ;       DR(LB)  :       Length of DR, bytes
;27561          ;       DR(LD)  :       Length of DR, digits
;27562          ;       DR(LDNLZ):      Length of DR, digits, no leading zeros
;27563          ;       [DV]    :       Current window on dividend, held in MTEMP8 -
;27564          ;                       MERRCOD
;27565          ;       DS.DIVP.REE.x:  Unpack reentry points, such as DS.DIVP.REE.1:
;27566          ;
;27567          ;       During init. check is done for divide-by-zero trap condition
;27568          ;       and DV=0 (Q_+0, CC_4) in that order.
;27569          ;
;27570          ;       Divisor (DR) is held in RTEMP8-RTEMP11, LSD to MSD, with
;27571          ;       the least signif. nibble in the sign nibble position, ie. the
;27572          ;       data is shifted right arithmetically one nibble.  Dividend
;27573          ;       Window ([DV]) is held in MTEMP8-MERRCOD, LSD to MSD, and is
;27574          ;       DR(LDNLZ) digits in length.
;27575          ;
;27576          ;       Input           GPR1            DR base address (address #1)
;27577          ;                       GPR3            DV base address (address #2)
;27578          ;                       GPR5            Q base address (address #3)
;27579          ;                       TEMP1           DR(LD) (length #1)
;27580          ;                       TEMP3           DV(LD) (length #2)
;27581          ;                       TEMP5           Destination length
;27582          ;                       Lengths         Checked for Reserved Operand
;27583          ;                                       Fault
```

```
                                                Resources       <Above, plus: >
                                                GPR0            Holds (PC-PCBACK) in byte#3
                                                                and pack/unpack data in other
                                                                bytes.  See banner on StateSave
                                                                code and Unpack routine.
                                                GPR2            (DR(LDNLZ)), pack/unpack data
                                                GPR4            (DR(LD)) in byte #2, pack/unpack
                                                                data
                                                TEMP0-TEMP3     Constructs [DV] during init.,
                                                                afterwhich...
                                                TEMP0           Holds Q digits for writes
                                                TEMP1           Holds DV bytes from memory
                                                TEMP2    <0>:   0=DR is negative, 1=DR is pos.
                                                TEMP3           General temp.
                                                TEMP4           Result of #Q digits that get
                                                                zero equation.  For any Q digit,
                                                                TEMP4+1 will tell ov (-), last
                                                                ov (0), or ok (+), providing
                                                                each digit (output or no) causes
                                                                TEMP4_TEMP4+1
                                                TEMP5           During init.: DV(LD),
                                                                Main: Q(LB)-1 (Q(LD)/2),
                                                                accounting for written bytes
                                                TEMP6           Init.: Q(LB)-1 (Q(LD)/2)
                                                                Main: General temp.
                                                TEMP7           General temp. for use with
                                                                LIT/LITRL
                                                MTEMP8-MERRCOD  [DV] holder, LSD to MSD
                                                                *M[ERRCOD] is not needed after
                                                                any BUS exception*
                                                FPDOFFSET       11. (after StateSave)
                                                FPD             1
                                                QREG            General temp.
                                                RTEMP8-RTEMP11  DR holder, LSD to MSD.
                                                RTEMP12         Misc. data
                                                RTEMP13         Ditto
                                                Stack           [DV] holder
                                                (SP)            Points to last byte of [DV] on
                                                                stack
                                                DREG            Gen. temp.
                                                MDR,WDR         I/O
                                                VA              Points to next Q byte (low add,
                                                                MSD end) after init. is done
                                                XB              Points to next byte of DV in
                                                                memory after init.
                                                PSL<C>          0 if dest. is even
                                                ALUS            Holds DR sign while reading
                                                                [DV] during init.
```

```
                                ;27632          ;                                   _1 if dest. is odd
                                ;27633          ;                       FLAG0       =0 if next Q digit is high
                                ;27634          ;                                   nibble for TEMP0
                                ;27635          ;                                   =1 if next Q digit is low nibble
                                ;27636          ;                                   for TEMP0
                                ;27637          ;                       FLAG1       <V> bit image:
                                ;27638          ;                                   _0 before first loop
                                ;27639          ;                                   _1 on ov (nonzero digit lost)
                                ;27640          ;                       FLAG2       <Z> bit image:
                                ;27641          ;                                   =1 before first loop
                                ;27642          ;                                   _0 if nonzero digit written
                                ;27643          ;                                   to destination
                                ;27644          ;                       FLAG3       =0 if this low nibble is
                                ;27645          ;                                   completing a byte with high
                                ;27646          ;                                   nibble zero (first Q byte
                                ;27647          ;                                   to be assembled)
                                ;27648          ;                                   =1 otherwise
                                ;27649          ;
                                ;27650          ;       Output          GPR0        0
                                ;27651          ;                       GPR1        Base address, source 1
                                ;27652          ;                       GPR2        0
                                ;27653          ;                       GPR3        Base address, source 2
                                ;27654          ;                       GPR4        0
                                ;27655          ;                       GPR5        Base address, destination
                                ;27656          ;                       PSL<NZVC>   C_0, NZV modified
                                ;27657          ;
                                ;27658          ;*******************************************************************************
```

```
                                ;27659 .TOC "  Decimal String                    :       Initialization"
                                ;27660
                                ;27661 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;27662
                                ;27663 DS.DIVP:
                                ;27664          ;--------------------------------;
U 11EC, 0586,CC37,0030,00A7,011E,E ;27665       M[FPDOFFSET]_CC_ZLITO[0]        ;FPDOFFSET_0, guess even dest.
                                ;27666
                                ;27667          ;--------------------------------;
U 11EE, 0CED,A592,4034,0047,011F,0 ;27668       R[R0]_M[PC],SET FPD             ;Get PC, figure & store PC-PCBACK
                                ;27669
                                ;27670          ;--------------------------------;
U 11F0, 0C85,9003,0034,0047,011F,2 ;27671       R[R0]_RB-M[PCBACK]
                                ;27672
                                ;27673          ;--------------------------------;
U 11F2, 0C84,02B7,0004,0047,011F,7 ;27674       R[R0]_RB.RR.8
                                ;27675
                                ;27676          ;--------------------------------;
U 11F7, 0480,05BE,42B1,4047,0122,2 ;27677       WB_R[TEMP5],WB<0>?              ;Is this an odd dest.?
                                ;27678
                                ;27679 =0
                                ;27680 DS.DIVP.IN1A:
                                ;27681          ;0-------------------------------;
                                ;27682          R[TEMP6]_M[TEMP5].SR.1,         ;L/2, destination (Q)
U 1222, 0C84,5591,8031,8047,011F,8 ;27683       NEXT/DS.DIVP.IN1
                                ;27684
                                ;27685          ;1-------------------------------;
                                ;27686          CC_ZLITO[1],                    ;Set C bit, odd destination
U 1223, 0180,0C37,0030,08A7,0122,2 ;27687       NEXT/DS.DIVP.IN1A
                                ;27688
                                ;27689
                                ;27690 DS.DIVP.IN1:
                                ;27691          ;--------------------------------;
                                ;27692          R[TEMP4]_M[TEMP1]               ;Adjust from xxxP6 code, copy
U 11F8, 0484,1592,4031,0047,011F,A ;27693                                       ;DR(LD)
                                ;27694
                                ;27695          ;--------------------------------;
U 11FA, 0484,3592,4031,4047,011F,C ;27696       R[TEMP5]_M[TEMP3]               ;Ditto, copy DV(LD)
                                ;27697
                                ;27698 DS.DIVP.REE.1:
                                ;27699          ;--------------------------------;
                                ;27700          STEPC_D_M[TEMP4].SR.1,          ;STEPC_D_L/2 for DS.READ
U 11FC, 0080,4001,A03D,8107,0109,1 ;27701       NEXT/DS.DIVP.IN2
```

```
                                ;27702 =000
                                ;27703          ;000-----------------------------;
                                ;27704          VA_D+R[R1]+1,PUSH,              ;No, or timer service, RTEMP8-RTEMP11_DR
U 1090, 0480,00A1,0034,44A7,053C,0  ;27705      NEXT/DS.READ.DIVP.DR
                                ;27706
                                ;27707 DS.DIVP.IN2:
                                ;27708          ;001-----------------------------;
                                ;27709          PUSH,INTPEND OR TIMER?,         ;See if something's pending, load GPR4
                                ;27710          NEXT/IE.SERV.IP.TS2,
U 1091, 0484,43B7,0AD5,0047,04F7,0  ;27711      R[R4]_M[TEMP4].RR.16
                                ;27712
                                ;27713          ;*********************************************************************
                                ;27714          ; RETURN+2 from DS.READ is exception, RETURN+3 is ok read.
                                ;27715          ; RETURN+4 is DR=0, we restore PC and take a 0 divisor trap.
                                ;27716          ;*********************************************************************
                                ;27717
                                ;27718 DS.DIVP.PCK.12.PRE:
                                ;27719          ;010-----------------------------;
U 1092, 0C84,05B7,0034,8047,0117,A  ;27720      R[R2]_0,NEXT/DS.DIVP.PCK.12     ;Zero GPR2 for future masking, pack up
                                ;27721
                                ;27722          ;011-----------------------------;
                                ;27723          PC_R[R3],                       ;Point PC for DV read
U 1093, 0080,05BE,4034,C487,011F,E  ;27724      NEXT/DS.DIVP.DRSGN
                                ;27725
                                ;27726          ;100-----------------------------;
                                ;27727          PC_ZEXT(M[TEMP0])+Q,SIZE[BYTE], ;DR=0, PSL<V> & FPD are ok, take trap
U 1094, 0480,0019,0000,0487,00FB,9  ;27728      NEXT/IE.FLT.DEC.DBZ
                                ;27729
                                ;27730          ;101-----------------------------;
                                ;27731          M[TEMP3]_ZLIT0[1],              ;Ints. pending from =001, pack up
U 1095, 0986,3C37,0030,0847,0109,2  ;27732      NEXT/DS.DIVP.PCK.12.PRE
                                ;27733 =
```

```
                                ;27734 DS.DIVP.DRSGN:
                                ;27735 DS.DIVP.REE.2:
                                ;27736          ;-------------------------------;
U 11FE, 0084,5711,8013,0047,0120,5  ;27737      R[TEMP12]_(M[TEMP5]+CONX(2)).SR.1;RTEMP12_DV(LB)
                                ;27738
                                ;27739
                                ;27740          ;*********************************************************************
                                ;27741          ; PC has DV base address (MSD)
                                ;27742          ; RTEMP12 has DV(LB)
                                ;27743          ; Load DREG with DR(LDNLZ)-1 (from GPR2) and read [DV].
                                ;27744          ;*********************************************************************
                                ;27745
                                ;27746          ;-------------------------------;
U 1205, 0880,0E7D,2034,8047,0109,9  ;27747      D_R[R2]-1
                                ;27748
                                ;27749 =000
                                ;27750 =001
                                ;27751          ;001----------------------------;
                                ;27752          R[TEMP7]_ZEXT(XB) PC_PC+1,      ;Get first/next byte from DV, is it
                                ;27753          NEXT/DS.DIVP.DVIN.1,            ;zero?
U 1099, 0C85,759E,4A41,E047,0122,4  ;27754      WX.NE.0?
                                ;27755
                                ;27756          ;*********************************************************************
                                ;27757          ; Here on WX.EQ.0? split from DS.DIVP.DVIN.1.  =010 is more left
                                ;27758          ; to DV string, check for ints. & timer service.  =011 is DV=0, and SIGN
                                ;27759          ; NIBBLE IS ZERO, INVALID SOURCE STRING... fix PC and IRD1.
                                ;27760          ;*********************************************************************
                                ;27761
                                ;27762 DS.DIVP.DVIN.INT:
                                ;27763          ;010----------------------------;
                                ;27764          PUSH,INTPEND OR TIMER?,         ;More left to string, check for ints.,
U 109A, 0480,0036,4AF0,0047,04F7,0  ;27765      NEXT/IE.SERV.IP.TS2             ;RETURN -1 for next byte or +4
                                ;27766
                                ;27767          ;011----------------------------;
                                ;27768          MTEMP0_R[R0].RR.24 Q_PCBACK,    ;TEMP0  (PC-PCBACK) in low byte,
                                ;27769          CLEAR FPD,                      ;Q_PCBACK, fix PC and IRD1
U 109B, 04E7,929C,7024,0047,0132,5  ;27770      NEXT/DS.DIVP.BAD
                                ;27771
                                ;27772 =101
                                ;27773          ;101----------------------------;
                                ;27774          M[TEMP3]_ZLIT0[2],              ;Load int. code 2, pack up
U 109D, 0186,3C37,0030,1047,0117,A  ;27775      NEXT/DS.DIVP.PCK.12
                                ;27776
                                ;27777          ;110----------------------------;
                                ;27778          M[TEMP3]_ZLIT0[2],              ;Load int. code 2, pack up
U 109E, 0186,3C37,0030,1047,0117,A  ;27779      NEXT/DS.DIVP.PCK.12
                                ;27780 =
```

```
                                ;27781          ;*************************************************************************
                                ;27782          ; Here on WX.NE.0? split from leading byte reads, DV
                                ;27783          ;*************************************************************************
                                ;27784 =0
                                ;27785 DS.DIVP.DVIN.1:
                                ;27786          ;0------------------------------;
                                ;27787          R[TEMP12]_RB-1,          ;Byte is zero, anymore bytes left in
U 1224, 0084,0E7D,0A33,0047,0909,A 378*  ;27788          WX.EQ.0?,NEXT/DS.DIVP.DVIN.INT  ;DV?
                                ;27789
                                ;27790          ;1------------------------------;
                                ;27791          R[TEMP12]_RB-1,                  ;Byte is non-zero, is it sign byte?,
U 1225, 0804,0E7D,0A33,0047,0922,6 378*  ;27792          CLEAR FLAG0,WX.EQ.0?             ;and clear FLAG0 for possible use later
                                ;27793
                                ;27794 =0
                                ;27795          ;0------------------------------;
                                ;27796          WB_M[TEMP7].AND.ZLIT0[0F0],      ;No, is this 1 digit (low nibble) or
U 1226, 0980,7C12,0A77,8047,0122,8       ;27797          WX.NE.0?,NEXT/DS.DIVP.DVIN.DIG   ;two?
                                ;27798
                                ;27799          ;1------------------------------;
                                ;27800          M[TEMP4]_ZLIT0[1],               ;Yes, DV is 1 digit + sign, we don't
U 1227, 0586,4C37,0030,0847,012F,7       ;27801          NEXT/DS.DIVP.TMPSPL              ;care if it is 0 + sign, load TEMP4 and
                                ;27802                                           ;enter MTEMP loading code
                                ;27803
                                ;27804
                                ;27805          ;*************************************************************************
                                ;27806          ; Nonzero, nonsign byte has been read.  RTEMP12 has # bytes left in DV
                                ;27807          ; AFTER this byte, DREG has GPR2-1.  =0 is 1 digit this byte (high nib
                                ;27808          ; zero), =1 is 2 digits.
                                ;27809          ;*************************************************************************
                                ;27810 =0
                                ;27811 DS.DIVP.DVIN.DIG:
                                ;27812          ;0------------------------------;
                                ;27813          M[TEMP4]_R[TEMP12].SL.1,         ;TEMP4_DV(LDNLZ), 1 digit this byte
U 1228, 0C86,45BD,C033,0047,012F,7       ;27814          NEXT/DS.DIVP.TMPSPL
                                ;27815
                                ;27816          ;1------------------------------;
U 1229, 0D86,4C37,0030,0847,0120,6       ;27817          M[TEMP4]_ZLIT0[1]                ;TEMP4_DV(LDNLZ), 2 digits this byte
                                ;27818
                                ;27819          ;------------------------------;
U 1206, 0086,45D1,0013,0047,012F,6       ;27820          M[TEMP4]_MB+(R[TEMP12].ASL.1)    ;
                                ;27821
                                ;27822          ;------------------------------;
U 12F6, 0880,0E71,2030,0047,012F,7       ;27823          D_D-1                            ;And fix DREG to account to 2 digits
                                ;27824                                           ;read already
```

```
                                        ;27825 DS.DIVP.TMPSPL:
                                        ;27826          ;--------------------------------;
U 12F7, 0980,0C30,06F0,3847,0916,5 377* ;27827          WB_D-ZLIT0[7.],WB<31-30>?        ;Now split to decide which MTEMP to
                                        ;27828                                           ;deposit this fraction of a longword
                                        ;27829                                           ;in.  DREG has DR(LDNLZ) MINUS THE
                                        ;27830                                           ;NUMBER OF DIGITS IN THE BYTE JUST READ.
                                        ;27831
                                        ;27832 =01
                                        ;27833          ;01--------------------------------;
                                        ;27834          WB_D-ZLIT0[15.],WB<31-30>?,      ;D>=7., we aren't in TEMP0
U 1165, 0D80,0C30,06F0,7847,0916,9 377* ;27835          NEXT/DS.D.VP.TMPSPL.1
                                        ;27836
                                        ;27837          ;11--------------------------------;
                                        ;27838          RNUM_Q_ZLIT0[0],                 ;No more than 1 longword total, store
U 1167, 0181,DC37,1030,0047,012F,8      ;27839          NEXT7DS.DIVP.MT1                 ;TEMP#
                                        ;27840
                                        ;27841 =01
                                        ;27842 DS.DIVP.TMPSPL.1:
                                        ;27843          ;01--------------------------------;
                                        ;27844          WB_D-ZLIT0[23.],WB<31-30>?,      ;D>=15., we aren't in TEMP1,
U 1169, 0580,0C30,06F0,B847,0916,D 377* ;27845          NEXT/DS.DIVP.TMPSPL.2            ;decide between TEMP2 or TEMP3
                                        ;27846
                                        ;27847          ;11--------------------------------;
                                        ;27848          RNUM_Q_ZLIT0[1],                 ;No more than 2 longwords total,
U 116B, 0D81,DC37,1030,0847,012F,9      ;27849          NEXT7DS.DIVP.MT2                 ;store TEMP#
                                        ;27850
                                        ;27851 =01
                                        ;27852 DS.DIVP.TMPSPL.2:
                                        ;27853          ;01--------------------------------;
                                        ;27854          RNUM_Q_ZLIT0[3],                 ;D>=23., we start in TEMP3,
U 116D, 0181,DC37,1030,1847,012F,B      ;27855          NEXT7DS.DIVP.DRDV                ;load MTEMP#
                                        ;27856
                                        ;27857          ;11--------------------------------;
                                        ;27858          RNUM_Q_ZLIT0[2],                 ;D<23., we are in TEMP2, load code
U 116F, 0D81,DC37,1030,1047,012F,A      ;27859          NEXT7DS.DIVP.MT3                 ;and zero TEMP3
                                        ;27860
                                        ;27861
                                        ;27862          ;*****************************************************************************
                                        ;27863          ; Zero unused high MTEMPs
                                        ;27864          ;*****************************************************************************
                                        ;27865
                                        ;27866 DS.DIVP.MT1:
                                        ;27867          ;--------------------------------;
U 12F8, 0986,1C37,0030,0047,012F,9      ;27868          M[TEMP1]_ZLIT0[0]
                                        ;27869
                                        ;27870 DS.DIVP.MT2:
                                        ;27871          ;--------------------------------;
U 12F9, 0986,2C37,0030,0047,012F,A      ;27872          M[TEMP2]_ZLIT0[0]
                                        ;27873
                                        ;27874 DS.DIVP.MT3:
                                        ;27875          ;--------------------------------;
U 12FA, 0586,3C37,0030,0047,012F,B      ;27876          M[TEMP3]_ZLIT0[0]
```

```
                                        ;27877          ;*************************************************************************
                                        ;27878          ; RNUM & QREG have the TEMP# to store the high fractional longword.
                                        ;27879          ; TEMP4 has DV(LDNLZ), GPR2 has DR(LDNLZ).
                                        ;27880          ;*************************************************************************
                                        ;27881
                                        ;27882 DS.DIVP.DRDV.
                                        ;27883          ;-------------------------------;
                                        ;27884          WB_M[TEMP4]-R[R2],              ;Is DV(LDNLZ) >= DR(LDNLZ)?
U 12FB, 0480,4000,06F4,8047,0917,0 338* ;27885          WB<31-30>?                      ;(split will be to =00 or =11)
                                        ;27886
                                        ;27887 =00
                                        ;27888          ;00-----------------------------;
                                        ;27889          WB_(D+CONX(1)).SR.1,WB<1-0>?,    ;Yes, split on <1-0> of total # digits
U 1170, 0C80,0731,8240,0047,0917,4 387* ;27890          NEXT/DS.DIVP.MT.FILL            ;to get
                                        ;27891
                                        ;27892
                                        ;27893          ;*************************************************************************
                                        ;27894          ; DV(LDNLZ) < DR(LDNLZ), Q will get +0, CC will get +0.  Case is of the
                                        ;27895          ; form:
                                        ;27896          ;
                                        ;27897          ;       xxxxx   T~~~~~~~~~~     ( Dividend < Divisor )
                                        ;27898          ;
                                        ;27899          ; FLAG0 was cleared at DS.DIVP.DVIN.1+1 when the first nonzero DV byte
                                        ;27900          ; was read.  TEMP4 (DV(LDNLZ)) will be loaded with GPR2 (DR(LDNLZ))
                                        ;27901          ; so the Q(extra) value will be one less than Q(LD).  The "zero high
                                        ;27902          ; dest. bytes" code will zero all of Q except for the sign byte (or
                                        ;27903          ; do nothing if Q(LD)=1 or 0) and the first & last subtraction will
                                        ;27904          ; produce a zero digit in the sign byte.  After loading TEMP4 &
                                        ;27905          ; setting FLAG0, we branch to DS.DIVP.MT2 to clear the upper TEMPs
                                        ;27906          ; and enter this code again.  FLAG0 being one indicates that the
                                        ;27907          ; TEMPs have been cleared.
                                        ;27908          ;*************************************************************************
                                        ;27909    =11
                                        ;27910          ;11-----------------------------;
U 1173, 0186,0C37,0430,0047,0122,A      ;27911          M[TEMP0]_ZLIT0[0],FLAG0?
                                        ;27912
                                        ;27913 =0
                                        ;27914          ;0------------------------------;
                                        ;27915          M[TEMP1]_ZLIT0[0],SET FLAG0,    ;Zero TEMP1, set flag, zero TEMP2
U 122A, 0946,1C37,0030,0047,012F,9      ;27916          NEXT/DS.DIVP.MT2                ;& TEMP3
                                        ;27917
                                        ;27918          ;1------------------------------;
U 122B, 0086,45BE,4034,8047,012F,C      ;27919          M[TEMP4]_R[R2]                  ;TEMP's are clear, DV(LDNLZ)_DR(LDNLZ)
                                        ;27920
                                        ;27921          ;-------------------------------;
                                        ;27922          R[TEMP13]_0,                    ;Clear for packing if interrupted.
U 12FC, 0084,05B7,0033,4047,0122,D      ;27923          NEXT/DS.DIVP.PRE.STACK
```

```
                                ;27924          ;*********************************************************************
                                ;27925          ; DREG has # of digits to get after TEMP7
                                ;27926          ; TEMP7 has byte read in already
                                ;27927          ; RNUM and QREG has TEMP# for the longword fraction
                                ;27928          ; Unused TEMPs have zero loaded
                                ;27929          ; PC points to next DV byte
                                ;27930          ; RTEMP12 has number of bytes left in DV
                                ;27931          ; Here on <1-0> split on total # of digits to get in [DV] for longword
                                ;27932          ; fractional alignment.
                                ;27933          ;*********************************************************************
                                ;27934 =00
                                ;27935 DS.DIVP.MT.FILL:
                                ;27936          ;00-----------------------------;
                                ;27937          M[TEMP.R]_R[TEMP7].RR.8,        ;Byte left, load it into low
U 1174, 0887,02B7,0B01,D8E7,010C,0 ;27938       IP.TS?,NEXT/DS.DIVP.MT.LOAD     ;byte, anything pending?
                                ;27939
                                ;27940          ;01-----------------------------;
                                ;27941          R[TEMP.R]_M[TEMP7].RR.16,       ;Word left, shift this byte around
U 1175, 0484,73B7,001C,0047,0117,8 ;27942       NEXT/DS.DIVP.MT.FILL.W
                                ;27943
                                ;27944          ;10-----------------------------;
                                ;27945          R[TEMP.R]_M[TEMP7].RR.24,       ;3bytes left, shift this byte around
U 1176, 0C84,73B7,002C,0047,0101,2 ;27946       NEXT/DS.DIVP.MT.FILL.3B
                                ;27947
                                ;27948          ;11-----------------------------;
                                ;27949          M[TEMP.R]_R[TEMP7],             ;Longword is high fraction, hold onto
U 1177, 0487,05BE,4031,C047,0106,2 ;27950       NEXT/DS.DIVP.MT.FILL.L          ;this byte
                                ;27951
                                ;27952
                                ;27953          ;*********************************************************************
                                ;27954          ; Continuation of word fraction read
                                ;27955          ;*********************************************************************
                                ;27956 =00
                                ;27957 DS.DIVP.MT.FILL.W:
                                ;27958          ;00-----------------------------;
                                ;27959          R[TEMP12]_RB-1,PUSH,            ;We're reading another byte of DV,
U 1178, 0084,0E7D,0033,0047,051B,A ;27960       NEXT/DS.SR.18                   ;read next byte, patch in and and
                                ;27961                                          ;load current TEMP
                                ;27962
                                ;27963          ;01-----------------------------;
                                ;27964          D_D-CONX(2),IP.TS?,             ;RETURN+1 is ok, TEMP has word,
U 1179, 0880,0730,2B10,18E7,010C,0 ;27965       NEXT/DS.DIVP.MT.LOAD            ;split on something pending, account
                                ;27966                                          ;for 2 more digits read
                                ;27967
                                ;27968 DS.DIVP.PCK.12:
                                ;27969          ;10-----------------------------;
                                ;27970          R[R0]_RB.OR.(M[TEMP3].RR.16),   ;RETURN+2 is exception, code is
U 117A, 0884,33BE,4014,0047,0124,C ;27971       NEXT/DS.DIVP.PCK.12.CON         ;loaded,mask into GPR0 and pack up
                                ;27972 =
```

```
                                ;27973          ;*****************************************************************************
                                ;27974          ; Continuation of 3 byte fraction fill
                                ;27975          ;*****************************************************************************
                                ;27976 =0**
                                ;27977 DS.DIVP.MT.FILL.3B:
                                ;27978          :0**-------------------------------;
                                ;27979          R[TEMP13]_ZEXT(X8) PC_PC+2,     ;Read next word
U 1012. 0485,759E,4013,6047,012F,D  ;27980      NEXT/DS.DIVP.MT.FILL.3B.1
                                ;27981
                                ;27982          ;1**-------------------------------;
                                ;27983          M[TEMP3]_ZLIT0[2],              ;BUS exception, pack up, code #2
U 1016, 0186,3C37,0030,1047,0117,A  ;27984      NEXT/DS.DIVP.PCK.12
                                ;27985
                                ;27986
                                ;27987 DS.DIVP.MT.FILL.3B.1:
                                ;27988          ;-------------------------------;
U 12FD, 0887,0292,4013,4047,012F,E  ;27989      M[TEMP.R]_MB.OR.(R[TEMP13].RR.16)
                                ;27990
                                ;27991          ;-------------------------------;
U 12FE, 0484,073C,0013,0047,012F,F  ;27992      R[TEMP12]_RB-CONX(2)            ;We read 2 more bytes of DV
                                ;27993
                                ;27994          ;-------------------------------;
                                ;27995          D_D-CONX(4),IP.TS?,             ;and 4 more digits, any ints.
U 12FF, 0880,0730,2B20,18E7,010C,0  ;27996      NEXT/DS.DIVP.MT.LOAD            ;pending?
```

```
                                ;27997          ;*************************************************************************
                                ;27998          ; Continuation of high longword fraction read
                                ;27999          ;*************************************************************************
                                ;28000 =0**
                                ;28001 DS.DIVP.MT.FILL.L:
                                ;28002          ;0**-------------------------------;
                                ;28003          R[TEMP13]_ZEXT(XB) PC_PC+2,     ;Read in next WORD
U 1062, 0485,759E,4013,6047,0117,C ;28004       NEXT/DS.DIVP.MT.FILL.L.1
                                ;28005
                                ;28006          ;1**-------------------------------;
                                ;28007          M[TEMP3]_ZLITO[2],              ;BUS exception, pack up (code #2)
U 1066, 0186,3C37,0030,1047,0117,A ;28008       NEXT/DS.DIVP.PCK.12
                                ;28009
                                ;28010 =00
                                ;28011 DS.DIVP.MT.FILL.L.1:
                                ;28012          ;00-------------------------------;
                                ;28013          PUSH,NEXT/DS.SR.18,             ;Patch byte and word together, read &
U 117C, 0087,0292,4023,4047,051B,A ;28014       M[TEMP.R]_MB.OR.(R[TEMP13].RR.24)       ;mask in next byte, RET +1 or +2
                                ;28015
                                ;28016          ;01-------------------------------;
                                ;28017          R[TEMP12]_RB-CONX(2),            ;We read 3 more bytes from DV (-1 done
U 117D, 0C84,073C,0013,0047,0130,0 ;28018       NEXT/DS.DIVP.MT.FILL.L1          ;2 cycles later)
                                ;28019
                                ;28020          ;10-------------------------------;
                                ;28021          R[R0]_RB.OR.(M[TEMP3].RR.16),    ;BUS exception from read, int.
U 117E, 0884,33BE,4014,0047,0124,C ;28022       NEXT/DS.DIVP.PCK.12.CON          ;code #2 is loaded, pack up
                                ;28023 =
                                ;28024
                                ;28025 DS.DIVP.MT.FILL.L1:
                                ;28026          ;--------------------------------;
U 1300, 0180,0C30,2030,3047,0130,1 ;28027       D_D-ZLITO[6]                     ;3 more bytes=6 more digits
                                ;28028
                                ;28029          ;--------------------------------;
U 1301, 0C84,0E7D,0B33,18E7,010C,0 ;28030       R[TEMP12]_RB-1,IP.TS?            ;Anything pending?
```

```
                                        ;28031          ;*************************************************************************
                                        ;28032          ; Here from high fraction read on IP.TS? split, or from reading 1 of 3
                                        ;28033          ; (maximum) next LONGWORDs to DS.DIVP.MT.LOAD: directly.
                                        ;28034          ;*************************************************************************
                                        ;28035 =000
                                        ;28036 DS.DIVP.MT.LOAD:
                                        ;28037          ;000--------------------------;
                                        ;28038          RNUM_Q_Q-1,WB<5-0>?,            ;Nothing pending or RETURN-1,
U 10C0, 0881,D0BB,123D,8047,0907,3 400* ;28039          NEXT7DS.DIVP.MT.LOAD.SPL        ;Back down 1 MTEMP and see if done
                                        ;28040
                                        ;28041 DS.DIVP.MT.LOAD.INT:
                                        ;28042          ;001--------------------------;
                                        ;28043          PUSH,INTPEND OR TIMER?,         ;Something pending, RETURN -1 or +4
U 10C1, 0480,0036,4AF0,0047,04F7,0      ;28044          NEXT/IE.SERV.IP.TS2
                                        ;28045
                                        ;28046 =101
                                        ;28047          ;101--------------------------;
                                        ;28048          M[TEMP3]_ZLIT0[2],              ;Interrupt code #2, pack up
U 10C5, 0186,3C37,0030,1047,0117,A      ;28049          NEXT/DS.DIVP.PCK.12
                                        ;28050 =
                                        ;28051
                                        ;28052          ;*************************************************************************
                                        ;28053          ; =110011 is read & store next LONGWORD.
                                        ;28054          ; =110111 is BUS exception, pack up (from =110011 split).
                                        ;28055          ; =111111 is RNUM & QREG went negative, we are done loading [DV].
                                        ;28056          ;*************************************************************************
                                        ;28057 =110011
                                        ;28058 DS.DIVP.MT.LOAD.SPL:
                                        ;28059          ;110011-----------------------;
                                        ;28060          R[TEMP.R]_XB PC_PC+4,
U 1073, 0885,7592,402C,2047,0130,2      ;28061          NEXT/DS.DIVP.MT.LOAD.LW
                                        ;28062
                                        ;28063          ;110111-----------------------;
                                        ;28064          M[TEMP3]_ZLIT0[2],              ;BUS exception, pack up, int. code #2
U 1077, 0186,3C37,0030,1047,0117,A      ;28065          NEXT/DS.DIVP.PCK.12
                                        ;28066
                                        ;28067 =111111
                                        ;28068          ;111111-----------------------;
                                        ;28069          R[TEMP13]_M[TEMP0].CLR3B,       ;RTEMP13_last [DV] byte read
U 107F, 0484,0537,0033,4047,0130,4      ;28070          NEXT/DS.DIVP.DV.SHF
                                        ;28071
                                        ;28072
                                        ;28073 DS.DIVP.MT.LOAD.LW:
                                        ;28074          ;--------------------------;
U 1302, 0180,0C30,2030,4047,0130,3      ;28075          D_D-ZLIT0[8.]                   ;8 less digits left (unless sign lw.)
                                        ;28076
                                        ;28077          ;--------------------------;
                                        ;28078          R[TEMP12]_RB-CONX(4),            ;4 less bytes left in DV, loop back
U 1303, 0C84,073C,0023,0047,010C,0      ;28079          NEXT/DS.DIVP.MT.LOAD
```

```
                                ;28080          ;*************************************************************************
                                ;28081          ; RTEMP13 has last byte read in high byte, zero in low 3 bytes.
                                ;28082          ;*************************************************************************
                                ;28083
                                ;28084 DS.DIVP.DV.SHF:
                                ;28085          ;-------------------------------;
U 1304, 0480,05BE,4A33,0047,0918,0 322*         ;28086          WB_R[TEMP12],WX.EQ.0?           ;Was all of DV read?
                                ;28087
                                ;28088 =00
                                ;28089          ;00-------------------------------;
                                ;28090          WB_D,WX.EQ.0?,                  ;No, DREG is zero if we read exact amt.,
U 1180, 0C80,0022,4A3D,8047,0122,C              ;28091          NEXT/DS.DIVP.DIGSHF             ;-1 if we read 1 more than needed
                                ;28092
                                ;28093
                                ;28094          ;*************************************************************************
                                ;28095          ; Enter below:
                                ;28096          ; 1) At DS.DIVP.MT.SHIFT: because all of DV was read.  Shift out
                                ;28097          ;    sign nibble.
                                ;28098          ; 2) At DS.DIVP.DIGSHF: from =00 split above.  All of DV was not read,
                                ;28099          ;    =0 is 1 more digit was read than desired, shift out low nibble,
                                ;28100          ;    =1 is [DV] is fine as is, store it on stack.
                                ;28101          ;*************************************************************************
                                ;28102 =01
                                ;28103 DS.DIVP.MT.SHIFT:
                                ;28104          ;01-------------------------------;
                                ;28105          PUSH,NEXT/DS.DIV10.DIVP,        ;Divide TEMP0-TEMP3 by 10.
U 1181, 0186,7C37,0030,0047,052B,4              ;28106          M[TEMP7]_ZLIT0[0]
                                ;28107
                                ;28108          ;10-------------------------------;
                                ;28109          VA_R[SP],                       ;Prepare to store [DV] on stack
U 1182, 0C80,05BE,4037,84A7,0130,5              ;28110          NEXT/DS.DIVP.PRE.STACK.1
                                ;28111 =
                                ;28112
                                ;28113 =0
                                ;28114 DS.DIVP.DIGSHF:
                                ;28115          ;0--------------------------------;
                                ;28116          PUSH,NEXT/DS.DIV10.DIVP,        ;Divide TEMP0-TEMP3 by 10, RETURN +1
U 122C, 0186,7C37,0030,0047,052B,4              ;28117          M[TEMP7]_ZLIT0[0]
                                ;28118
                                ;28119 DS.DIVP.PRE.STACK:
                                ;28120          ;1--------------------------------;
U 122D, 0C80,05BE,4037,84A7,0130,5              ;28121          VA_R[SP]                        ;TEMPs are shifted, VA_top of stack
                                ;28122
                                ;28123 DS.DIVP.PRE.STACK.1:
                                ;28124          ;-------------------------------;
U 1305, 0886,B5BE,4030,C047,0130,6              ;28125          M[ERRCOD]_R[TEMP3]               ;Move [DV] to high MTEMPs
                                ;28126
                                ;28127          ;-------------------------------;
U 1306, 0086,A5BE,4030,8047,0130,7              ;28128          M[TEMP10]_R[TEMP2]
                                ;28129
                                ;28130          ;-------------------------------;
U 1307, 0086,95BE,4030,4047,0130,8              ;28131          M[TEMP9]_R[TEMP1]
                                ;28132
                                ;28133          ;-------------------------------;
U 1308, 0886,85BE,4B70,1847,0118,5              ;28134          M[TEMP8]_R[TEMP0],ALUS?         ;What is DR sign?
```

```
                                ;28135 .TOC "  Decimal String                    :       Storing Dividend on Stack"
                                ;28136
                                ;28137          ;*************************************************************************
                                ;28138          ; We move up stack workspace area while moving down MTEMPs.  Loop will
                                ;28139          ; exit when RNUM (QREG)=7.
                                ;28140          ;*************************************************************************
                                ;28141 =01
                                ;28142          ;01-----------------------------;
                                ;28143          M[TEMP2]_ZLIT0[1],              ;DR is positive
U 1185, 0586,2C37,0030,0847,0130,9 ;28144       NEXT/DS.DIVP.STACK
                                ;28145
                                ;28146          ;11-----------------------------;
U 1187, 0186,2C37,0030,0047,0130,9 ;28147       M[TEMP2]_ZLIT0[0]               ;DR is negative
                                ;28148
                                ;28149 DS.DIVP.STACK:
                                ;28150          ;-------------------------------;
U 1309, 0886,12B7,0023,4047,0130,A ;28151       M[TEMP1]_R[TEMP13].RR.24        ;TEMP1_last byte read into low byte
                                ;28152
                                ;28153          ;-------------------------------;
                                ;28154          VA_D_M[VA]-ZLIT0[16.],          ;Back up to beginning of workspace area
U 130A, 0D81,BC10,2030,84A7,0101,0 ;28155       NEXT?DS.DIVP.STACK.INT          ;(to DREG for SP setting)
                                ;28156
                                ;28157 DS.DIVP.STACK.LOP:
                                ;28158          ;-------------------------------;
                                ;28159          VA_VA+4,RNUM_Q_Q-1,             ;Advance up stack, MTEMPs, and see
U 130B, 0081,D0B8,123D,8447,0900,7 400* ;28160  WB<5-0>?,NEXT/DS.DIVP.STACK.WR   ;if done (WB=7)
                                ;28161
                                ;28162 =00000
                                ;28163 =00111
                                ;28164 DS.DIVP.STACK.WR:
                                ;28165          ;00111----------------------------;
                                ;28166          R[SP]_D,IP.TS?,                 ;Point SP past stack workspace area,
U 1007, 0C84,05B2,4B37,98E7,010D,8 ;28167       NEXT/DS.DIVP.QFIX               ;split on ints. pending
                                ;28168
                                ;28169 =01111
                                ;28170          ;01111----------------------------;
                                ;28171          WRITE M[TEMP.R],SIZE[LONG],     ;Nothing pending, loop back -OR- not
U 100F, 0081,0592,4020,05D8,0130,B ;28172       NEXT/DS.DIVP.STACK.LOP          ;finished writing stack, loop back
                                ;28173
                                ;28174 DS.DIVP.STACK.INT:
                                ;28175          ;10000----------------------------;
                                ;28176          RNUM_Q_ZLIT0[11.],              ;Start with MERRCOD, check for ints.
                                ;28177          PUSH,INTPEND OR TIMER?,         ;before writing to stack, RETURN
U 1010, 0D81,DC37,1AF0,5847,04F7,0 ;28178       NEXT/IE.SERV.IP.TS2             ;-1 or +4
                                ;28179
                                ;28180 =10011
                                ;28181          ;10011----------------------------;
                                ;28182          M[TEMP3]_ZLIT0[2],              ;BUS exception, pack up, code #2
U 1013, 0186,3C37,0030,1047,0117,A ;28183       NEXT/DS.DIVP.PCK.12
                                ;28184
                                ;28185 =10100
                                ;28186          ;10100----------------------------;
                                ;28187          M[TEMP3]_ZLIT0[2],              ;Ints. pending, pack up, code #2
U 1014, 0186,3C37,0030,1047,0117,A ;28188       NEXT/DS.DIVP.PCK.12
                                ;28189 =
```

```
                                ;28190 .TOC     "       Decimal String                  :       Zeroing high dest. bytes"
                                ;28191
                                ;28192          ;*************************************************************************
                                ;28193          ; Here on IP.TS? split after saving [DV] on Stack.  We calculate the
                                ;28194          ; high number of Q digits to get zeroed via the following algorithm:
                                ;28195          ;
                                ;28196          ; Q(extra) = Q(LD) - DV(LDNLZ) + DR(LDNLZ) - 1
                                ;28197          ;
                                ;28198          ; If Q(extra) is > 0, Q(extra) is the number of high (most significant)
                                ;28199          ; digits to get zero'd (ie, extra digits).  If = 0, nothing is done.
                                ;28200          ; If < 0, Q(extra) is the number of output digits to be swallowed by
                                ;28201          ; the DIVP main loop before beginning to write the infinite precision
                                ;28202          ; least significant digits of the quotient.
                                ;28203          ;*************************************************************************
                                ;28204 =000
                                ;28205 DS.DIVP.QFIX:
                                ;28206 DS.DIVP.REE.3:
                                ;28207          ;000--------------------------------;
U 10D8, 0880,05BE,4035,44A7,0130,C  ;28208          VA_R[R5],NEXT/DS.DIVP.QFIX.1    ;VA_Q MSD address
                                ;28209
                                ;28210          ;001--------------------------------;
                                ;28211          PUSH,INTPEND OR TIMER?,         ;Something pending, RETURN-1 or +4
                                ;28212          NEXT/IE.SERV.IP.TS2,
U 10D9, 0D86,3C37,0AF0,1847,04F7,0  ;28213          M[TEMP3]_ZLIT0[3]
                                ;28214
                                ;28215 =101
                                ;28216          ;101--------------------------------;
                                ;28217          R[R4]_RB.OR.(M[TEMP6].RR.8),    ;TEMP3 has int. code#3, load
U 10DD, 0084,63BE,4005,0047,0132,A  ;28218          NEXT/DS.DIVP.PCK.34             ;TEMP6, finish packing
                                ;28219 =
```

```
                                        ;28220          ;*************************************************************************
                                        ;28221          ; Q(LD) is ( (TEMP6.SL)+PSL<C> ), and DV(LDNLZ) is (TEMP4).  If PSL<C>
                                        ;28222          ; is =1, the +PSL<C> in the calc. of Q(LD) and the -1 in the Q(extra)
                                        ;28223          ; equation will cancel out.  If PSL<C> is =0, we do the -1 in the
                                        ;28224          ; Q(extra) computation.
                                        ;28225          ; If Q(extra) is >= 0, we set flags based on PSL<C> (=0 if even dest.,
                                        ;28226          ; =1 if odd dest.) since the first Q digit placement will depend on
                                        ;28227          ; Q(LD) evenness/oddness.  If Q(extra) is <0, we set flags based on:
                                        ;28228          ;       Q(extra) + ( Q(LD)<0> )          [Q(LD)<0> is PSL<C>]
                                        ;28229          ; This value will indicate in which nib. Q digits start (high/low).  If
                                        ;28230          ; this result is even, we have an even Q(extra) and Q(LD) OR an odd
                                        ;28231          ; Q(extra) and Q(LD), in which case Q digits begin in the low nibble.
                                        ;28232          ; If the result is odd, we have an odd/even or even/odd pair of Q(extra)
                                        ;28233          ; and Q(LD), and the first Q digit goes in the high nibble of the first
                                        ;28234          ; byte.  **Note that the case of DV(LDNLZ)<DR(LDNLZ) caused
                                        ;28235          ; TEMP4_GRP2, so Q(extra) will get Q(LD)-1 (for even or odd Q(LD)).
                                        ;28236          ;*************************************************************************
                                        ;28237
                                        ;28238 DS.DIVP.QFIX.1:
                                        ;28239          ;-------------------------------;
                                        ;28240          M[TEMP4]_(R[TEMP6].ASL.1)-MB,   ;Do partial Q(LD)-DV(LDNLZ)
U 130C, 0086,45D3,0951,8047,0122,E      ;28241          PSL<C>?
                                        ;28242
                                        ;28243 =0
                                        ;28244          ;0-------------------------------;
U 122E, 0586,4C10,0030,0847,0122,F      ;28245          M[TEMP4]_MB-ZLIT0[1]             ;+0 -1.. (Even Q(LD))
                                        ;28246
                                        ;28247          ;1-------------------------------;
                                        ;28248          M[TEMP4]_MB+R[R2],SIGND CMP?,    ;(Odd Q(LD))..Make Q(extra) by adding
U 122F, 0486,4001,0B64,8047,0918,9 374* ;28249          SIZE[LONG]                       ;DR(LDNLZ), what is sign?
                                        ;28250
                                        ;28251 =01
                                        ;28252          ;01------------------------------;
U 1189, 0080,0036,4970,0047,0123,0      ;28253          PSL<C>?,NEXT/DS.DIVP.QFIX.FLAGS ;>=0, split on evenness - oddness of
                                        ;28254                                           ;Q(LD)
                                        ;28255
                                        ;28256          ;11------------------------------;
                                        ;28257          WB_M[TEMP4]+R[ZERO]+PSLC,        ;Q(extra) + (0 if even dest., 1 if odd
U 118B, 0880,40C1,028D,8047,0923,0 344* ;28258          WB<0>?                           ;dest.)..is result odd/even?
                                        ;28259
                                        ;28260 =0
                                        ;28261 DS.DIVP.QFIX.FLAGS:
                                        ;28262          ;0-------------------------------;
                                        ;28263          FLAGS_ZLIT0[5],                  ;First Q digit goes in low nibble of
U 1230, 0980,0C37,0030,2B07,0130,D      ;28264          NEXT/DS.DIVP.QFIX.SPL            ;first Q byte, set flags accordingly &
                                        ;28265                                           ;init. <ZV> images
                                        ;28266
                                        ;28267          ;1-------------------------------;
U 1231, 0980,0C37,0030,6307,0130,D      ;28268          FLAGS_ZLIT0[0C]                  ;First Q digit goes in high nibble of
                                        ;28269                                           ;first Q byte, set flags accordingly &
                                        ;28270                                           ;init. <ZV> images
```

```
                                ;28271          ;*****************************************************************************
                                ;28272          ; FLAGS are set, TEMP4 has Q(extra)
                                ;28273          ;*****************************************************************************
                                ;28274
                                ;28275 DS.DIVP.QFIX.SPL:
                                ;28276 DS.DIVP.REE.4:
                                ;28277          ;-------------------------------;
                                ;28278          WB_R[TEMP4],SIZE[LONG],         ;What is sign of Q(extra)?
U 130D, 0C80,05BE,4B61,0047,0906.8 351* ;28279  SIGND CMP?,NEXT/DS.DIVP.QFIX.FIL
                                ;28280
                                ;28281 =0000
                                ;28282 =0111
                                ;28283          ;0111-----------------------------;
                                ;28284          M[TEMP6]_MB-ZLIT0[1],FLAG3?,    ;RETURN-1 from write/int. check,
U 1067, 0586,6C10,04F0,0847,0123,2      ;28285  NEXT/DS.DIVP.QFIX.FIL.CK        ;(Q(LB)-1)_(Q(LB)-1)-1 (another byte
                                ;28286                                          ;written), what is digit/nib. flag?
                                ;28287
                                ;28288
                                ;28289          ;*****************************************************************************
                                ;28290          ; Here on SIGND CMP? split.  If <= zero, fine, else write high zero
                                ;28291          ; Q-bytes checking for ints. with each write.
                                ;28292          ;*****************************************************************************
                                ;28293 =1000
                                ;28294 DS.DIVP.QFIX.FIL:
                                ;28295          ;1000-----------------------------;
                                ;28296          PUSH,SIZE[BYTE],WRITE ZLIT0[0], ;Q(extra) is (still) >0, write
                                ;28297          INTPEND OR TIMER?,              ;zeroes, check for ints., return
U 1068, 0D80,0C37,0AC0,05D8,04F7,0      ;28298  NEXT/IE.SERV.IP.TS2             ;-1 or +4
                                ;28299
                                ;28300 DS.DIVP.QFIX.PRE.SS:
                                ;28301          ;1001-----------------------------;
                                ;28302          M[TEMP5]_R[TEMP6],              ;Ok, load TEMP5 with remaining Q(LB)-1
U 1069, 0C86,55BE,4031,8047,0130,F      ;28303  NEXT/DS.DIVP.SS
                                ;28304
                                ;28305          ;1010-----------------------------;
                                ;28306          M[TEMP5]_R[TEMP6],              ;Ok, load TEMP5 with remaining Q(LB)-1
U 106A, 0C86,55BE,4031,8047,0130,F      ;28307  NEXT/DS.DIVP.SS
                                ;28308
                                ;28309 =1100
                                ;28310          ;1100-----------------------------;
                                ;28311          M[TEMP3]_ZLIT0[4],              ;BUS exception/ints. pending,
U 106C, 0586,3C37,0030,2047,0132,6      ;28312  NEXT/DS.DIVP.PCK.4              ;pack up, int. code #4
                                ;28313 =
```

```
                                        ;28314          ;*************************************************************************
                                        ;28315          ; Here after writing a zero byte to high Q on FLAG3? split.
                                        ;28316          ; =0 is we wrote 1 digit to Q (first byte, even dest.), =1 is
                                        ;28317          ; 2 digit byte.
                                        ;28318          ;*************************************************************************
                                        ;28319 =0
                                        ;28320 DS.DIVP.QFIX.FIL.CK:
                                        ;28321          ;0-----------------------------;
                                        ;28322          SET FLAG3,M[TEMP4]_MB-1,        ;All future bytes will be 2 digits,
                                        ;28323          SIGND CMP?,SIZE[LONG],          ;dec. count by 1 digit, what is sign?
U 1232, 045E,4080,0B6D,8047,0918,C 380* ;28324          NEXT/DS.DIVP.QFIX.FIL.CK1
                                        ;28325
                                        ;28326          ;1-----------------------------;
                                        ;28327          M[TEMP4]_MB-ZLIT0[2],           ;2 digits in this byte, what is
U 1233, 0986,4C10,0B60,1047,0918,C 413* ;28328          SIZE[LONG],SIGND CMP DEF?
                                        ;28329
                                        ;28330 =00
                                        ;28331 DS.DIVP.QFIX.FIL.CK1:
                                        ;28332          ;00----------------------------;
                                        ;28333          VA_M[VA]+ZLIT0[1],              ;>0, more digits to zero fill
U 118C, 0D81,BC11,0030,0CA7,0106,8      ;28334          NEXT/DS.DIVP.QFIX.FIL
                                        ;28335
                                        ;28336          ;01----------------------------;
                                        ;28337          VA_M[VA]+ZLIT0[1],CLEAR FLAG0,  ;=0, all done, do StateSave
U 118D, 0D01,BC11,0030,0CA7,0106,9      ;28338          NEXT/DS.DIVP.QFIX.PRE.SS
                                        ;28339
                                        ;28340          ;10----------------------------;
U 118E, 011E,4C37,0030,0047,0130,E      ;28341          M[TEMP4]_ZLIT0[0],CLEAR FLAG3   ;<0, do NOT move on to next Q byte,
                                        ;28342                                          ;this Q byte has "1 Q digit in low
                                        ;28343                                          ;nibble", ie. next (first real) Q digit
                                        ;28344 =
                                        ;28345
                                        ;28346          ;----------------------------;
U 130E, 0846,5E7C,0031,8047,0130,F      ;28347          M[TEMP5]_R[TEMP6]+1,SET FLAG0   ;And copy Q(LB)-1 (plus 1) to TEMP5,
                                        ;28348                                          ;we really zeroed only 1/2 of a byte,
                                        ;28349                                          ;next Q digit goes in low nibble
```

```
;28350 .TOC     "       Decimal String                  :       StateSave"
;28351
;28352          ;*****************************************************************************
;28353          ;
;28354          ; On Unpacks, int. code 0 causes a StateSave unpack, ie. an unpack that
;28355          ; restores the state of the instruction before entering the divide
;28356          ; loops.  State is stored in GPR's & stack now -- no packing is done
;28357          ; in ints./exceptions.  TEMP6 is copied to TEMP5 before packing, and
;28358          ; FPDOFFSET_ZLITO[11.].
;28359          ;
;28360          ; Here with TEMP5 containing Q(LB)-1, accounting for any zero'd bytes
;28361          ; written to Q during initialization.
;28362          ;
;28363          ; A StateSave is:
;28364          ;
;28365          ;       Done already:
;28366          ;
;28367          ;       Stack:  (OldSP-1)                                       (OldSP-16)
;28368          ;                  ! MTEMP8 ! MTEMP9 ! MTEMP10 ! MERRCOD !
;28369          ;
;28370          ;       (SP)            User (SP)-16.
;28371          ;
;28372          ;       Done in StateSave (* means value is + or 0):
;28373          ;
;28374          ;       FPDOFFSET       ZLITO[11.]
;28375          ;
;28376          ;       GPR0    #0      FLAGS*
;28377          ;               #1      TEMP2*
;28378          ;               #2      Int. Code (zero)(0)
;28379          ;               #3      (PC-PCBACK)
;28380          ;
;28381          ;       GPR2    #0      (DR(LDNLZ))
;28382          ;               #1      (New) TEMP5*
;28383          ;               #2      TEMP4
;28384          ;               #3      TEMP1*
;28385          ;
;28386          ;       GPR4    #0      VA-GPR5*
;28387          ;               #1      PC(XB) - GPR3*
;28388          ;               #2      (DR(LD))
;28389          ;               #3      Unused, filled zero
;28390          ;
;28391          ;*****************************************************************************
```

```
                                        ;28392 DS.DIVP.SS:
                                        ;28393          ;--------------------------------;
U 130F, 0081,B000,2035,4047,0131,0      ;28394          D_M[VA]-R[R5]                   ;D_VA offset from GPR5
                                        ;28395
                                        ;28396          ;--------------------------------;
U 1310, 0084,0022,4035,0047,0119,0      ;28397          R[R4]_D.OR.RB                   ;GPR4_VA-GPR5
                                        ;28398
                                        ;28399 =00
                                        ;28400          ;00-------------------------------;
                                        ;28401          R[R0]_RB.OR.(M[TEMP2].RR.24),   ;GPR0_TEMP2 masked in byte#1,
U 1190, 0C84,23BE,4024,0047,053F,2      ;28402          PUSH,NEXT/DS.SR.GPR0IN.FLAGS    ;mask in FLAGS
                                        ;28403
                                        ;28404          ;01-------------------------------;
                                        ;28405          STEPC_14.,                      ;Load stpec for aborting main loop.
                                        ;28406          PUSH,NEXT/DS.SR.21,             ;Load GPR2, Get PC(XB) offset
U 1191, 00B7,A000,0034,C047,053E,4      ;28407          MTEMP0_PC-R[R3]
                                        ;28408
                                        ;28409          ;10-------------------------------;
                                        ;28410          DEC STEPC,                      ;STEPC_13
U 1192, 0C9C,03BE,4025,0047,0131,1      ;28411          R[R4]_RB.OR.(M[TEMP0].RR.24)    ;GPR4 is done
                                        ;28412 =
                                        ;28413
                                        ;28414 DS.DIVP.REE.0:
                                        ;28415          ;--------------------------------; Load stepc for aborting main loop.
                                        ;28416          DEC STEPC,                      ;STEPC_12
                                        ;28417          M[FPDOFFSET]_ZLIT0[11.],        ;Load FPDOFFSET for IE.PACK.DONE
U 1311, 099E,CC37,0B20,5847,0923,4 350* ;28418          COUNT OR INT_TIMER?,SIZE[LONG]  ;branch in IANDE code, is
                                        ;28419                                          ;anything pending?
```

```
                                ;28420 .TOC     "       Decimal String                  :       Main Loops"
                                ;28421
                                ;28422          ;*************************************************************************
                                ;28423          ; StateSave has been done, here on IP.TS? BUT.
                                ;28424          ;*************************************************************************
                                ;28425 =0
                                ;28426          ;0-----------------------------; STEPC_11. for aborting main loop
                                ;28427          M[TEMP0] ZLIT0[0],DEC STEPC,    ;Nothing pending or timer service,
U 1234, 0D9E,0C37,0030,0047,0119,5      ;28428  NEXT/DS.DIVP.ML.S               ;Zero Q digit holder and begin calc.
                                ;28429
                                ;28430          ;1-----------------------------;
                                ;28431          PUSH,INTPEND OR TIMER?,         ;Something pending, return -1 or
U 1235, 0480,0036,4AF0,0047,04F7,0      ;28432  NEXT/IE.SERV.IP.TS2             ;go to IE.PACK.DONE
                                ;28433
                                ;28434
                                ;28435          ;*************************************************************************
                                ;28436          ; Enter below 3 ways:
                                ;28437          ; 1) At DS.DIVP.ML.T0: on a FLAG0? split.
                                ;28438          ;       =00 is previous Q digit completed a byte, zero TEMP0.
                                ;28439          ;       =01 is previous Q digit was the high nibble, leave TEMP0 as is.
                                ;28440          ; 2) At DS.DIVP.ML.S: directly from the =0 branch taken after DIVP
                                ;28441          ;       StateSave.
                                ;28442          ; 3) At DS.DIVP.ML.S: on a ALU5? split after incrementing the proper
                                ;28443          ;       Q digit.
                                ;28444          ;       =01 is previous pass thru loop didn't BORROW, continue
                                ;28445          ;       subtracting until BORROW occurs.
                                ;28446          ;       =11 is previous pass thru loop created a BORROW, check for ints.
                                ;28447          ;       and restore [DV] to value before previous pass and decrement
                                ;28448          ;       appropriate Q digit.
                                ;28449          ;*************************************************************************
                                ;28450 =00
                                ;28451 DS.DIVP.ML.T0:
                                ;28452          ;00-----------------------------;
U 1194, 0D86,0C37,0030,0047,0119,5      ;28453  M[TEMP0]_ZLIT0[0]
                                ;28454
                                ;28455 DS.DIVP.ML.S:
                                ;28456          ;01-----------------------------;
                                ;28457          M[TEMP8] (MB-R[TEMP8]).BCD,     ;Move up 4 longwords, high digits are
U 1195, 0886,8000,4022,0047,0131,4      ;28458  NEXT/DS.DIVP.ML.S.L             ;zero if unused
                                ;28459
                                ;28460          ;10-----------------------------;
                                ;28461          M[TEMP8] (MB+R[TEMP8]).BCD,     ;Correct [DV]
U 1196, 0486,8001,4022,0047,0131,5      ;28462  NEXT/DS.DIVP.ML.S.CON
                                ;28463
                                ;28464          ;11-----------------------------;
                                ;28465          PUSH,INTPEND OR TIMER?,         ;Last pass was 1 too many, check for
U 1197, 0480,0036,4AF0,0047,04F7,0      ;28466  NEXT/IE.SERV.IP.TS2             ;ints, return -1 or never
```

```
                                ;28467 DS.DIVP.ML.S.L:
                                ;28468          ;--------------------------------;
                                ;28469          M[TEMP9]_(MB-R[TEMP9]-ALKC).BCD, ;CI_ALK<C> after first subtract
U 1314, 0C86,9040,4022,4047,00FF,2 ;28470       NEXT/DS.DIVP.ML.SKIP             ;
                                ;28471
                                ;28472 1312:
                                ;28473 QU.LOCK.CHECK:
                                ;28474          ;****** force address **********; ** NOT BEEN USED **
                                ;28475          ;10--------------------------------; SOFTWARE LOCK NOT SET
                                ;28476          WB_M[TEMP1].AND.ZLIT0[6],        ; CHECK (HEADER) FOR QUAD ALIGNMENT
U 1312, 0980,1C12,0A30,3047,00CF,8 ;28477       WX.EQ.0?,NEXT/QU.CHECK.ALIGN     ;
                                ;28478
                                ;28479
                                ;28480
                                ;28481 0FF2:
                                ;28482 DS.DIVP.ML.SKIP:
                                ;28483          ;**********FORCE ADDRESS*********;
                                ;28484          M[TEMP10]_(MB-R[TEMP10]-ALKC).BCD,
U 0FF2, 0C86,A040,4322,8047,007F,6 ;28485       DBZ STEPC?                       ;Fault if looped more than 11 times.
                                ;28486
                                ;28487 07F6:    ;0********FORCE ADDRESS*********;
                                ;28488          M[ERRCOD]_(MB-R[TEMP11]-ALKC).BCD,      ;Which Q digit are we?
                                ;28489          ALUS_UNSGN,FLAG0?,               ;
U 07F6, 0C86,B040,4422,D0C7,0123,6 ;28490       NEXT7DS.DIVP.ML.DIGIT            ;
                                ;28491
                                ;28492 07F7:    ;1********FORCE ADDRESS*********;
                                ;28493          M[ERRCOD]_OLIT16[120.],           ;
U 07F7, 0186,BF37,0033,C047,00FF,8 ;28494       NEXT/IE.OPER.FAULT               ; Illegal BCD digits, fault.
                                ;28495
                                ;28496
                                ;28497 1313:    ;****** force address **********; ** NOT BEEN USED**
                                ;28498          ;11--------------------------------; SOFTWARE INTERLOCK SET
                                ;28499          WB_M[TEMP1].AND.ZLIT0[6],        ; CHECK (HEADER) FOR QUAD ALIGNMENT
U 1313, 0980,1C12,0A30,3047,00CC,E ;28500       WX.EQ.0?,NEXT/QU.LOCKED.ALIGN    ; MAKE SURE IT'S NOT A BROKEN Q.
                                ;28501                                           ; EVEN IF IT IS LOCKED UP
                                ;28502
                                ;28503 =0
                                ;28504 DS.DIVP.ML.DIGIT:
                                ;28505          ;0--------------------------------;
U 1236, 0D86,0C11,0030,7847,0123,7 ;28506       M[TEMP0]_MB+ZLIT0[15.]           ; High nibble, set up to add 16.
                                ;28507                                           ; (10 hex), see if we sub'd too far
                                ;28508
                                ;28509          ;1--------------------------------;
                                ;28510          M[TEMP0]_MB+1,                   ; (Low nibble), add and see if we went
U 1237, 0086,0081,0B7D,9847,0119,5 ;28511       ALUS?,NEXT/DS.DIVP.ML.S          ; too far
                                ;28512
                                ;28513
                                ;28514          ;*****************************************************************************
                                ;28515          ; Here after adding sub. correction to MTEMP8.  After correcting
                                ;28516          ; the Q digit (decrement), if this byte is finished we decide
                                ;28517          ; what to do with it.
                                ;28518          ;*****************************************************************************
                                ;28519
                                ;28520 DS.DIVP.ML.S.CON:
                                ;28521          ;--------------------------------;
```

```
1315, 0086,9041,4022,4047,0131,6        ;28522          M[TEMP9]_(MB+R[TEMP9]+ALKC).BCD ;Add in carry from last addition
```

```
                                ;28523          ;--------------------------------;
U 1316, 0886,A041,4022,8047,0131,7  ;28524      M[TEMP10]_(MB+R[TEMP10]+ALKC).BCD
                                ;28525
                                ;28526          ;--------------------------------;
                                ;28527          M[ERRCOD]_(MB+R[TEMP11]+ALKC).BCD,      ;Which Q digit are we?
U 1317, 0C86,B041,4422,C047,0123,8  ;28528      FLAG0?
                                ;28529
                                ;28530 =0
                                ;28531          ;0-------------------------------;
                                ;28532          M[TEMP0]_MB-ZLIT0[010],         ;High nibble, set nibble flag for low
                                ;28533          SET FLAG0,                      ;nibble next time
U 1238, 0146,0C10,0030,8047,0131,8  ;28534      NEXT/DS.DIVP.ML.S.HIG
                                ;28535
                                ;28536          ;1-------------------------------;
                                ;28537          M[TEMP0]_MB-ZLIT0[01],FLAG3?,   ;We are low nibble, this byte is
U 1239, 0506,0C10,04F0,0847,0123,C  ;28538      NEXT/DS.DIVP.ML.Q,CLEAR FLAG0   ;finished, set nibble flag for high
                                ;28539                                          ;nibble next time and split on # of
                                ;28540                                          ;digits in this byte (=1 if first byte,
                                ;28541                                          ;even dest.)
                                ;28542
                                ;28543
                                ;28544          ;*****************************************************************************
                                ;28545          ; Enter at DS.DIVP.ML.S.HIG: after fixing TEMP0 & clearing FLAG0
                                ;28546          ; on a high nibble.  If TEMP5>0 keep looping, else if TEMP4>=-1
                                ;28547          ; (Q(extra)>=-1), we are done, this last high nibble is really the
                                ;28548          ; last high nibble.  The TEMP4 check is made for the case of
                                ;28549          ; Q(LD)=1 or 0, and Q(extra) < 0.
                                ;28550          ;*****************************************************************************
                                ;28551
                                ;28552 DS.DIVP.ML.S.HIG:
                                ;28553          ;--------------------------------;
U 1318, 0880,5592,4A30,0047,0923,A 322*  ;28554  WB_M[TEMP5],WX.EQ.0?             ;Is this the last Q byte?
                                ;28555
                                ;28556 =0
                                ;28557          ;0-------------------------------;
                                ;28558          R[TEMP1]_(RB M[TEMP1]).RL.4,    ;No, regardless of Q(extra) we keep
U 1234, 0C84,1237,0030,4047,0131,B  ;28559      NEXT/DS.DIVP.ML.SHIFT           ;looping
                                ;28560
                                ;28561          ;1-------------------------------;
                                ;28562          WB_M[TEMP4]+ZLIT0[1],           ;Yes, are we swallowing extra Q digits,
U 123B, 0180,4C11,06F0,0847,0919,9 377*  ;28563  WB<31-30>?                      ;and if so is this high nibble the last
                                ;28564                                          ;to be swallowed (Q(extra)=-1)?
                                ;28565 =01
                                ;28566          ;01------------------------------;
                                ;28567          ALUS_BCD SIGN,ZERO(M[TEMP1]),   ;Q(extra) >= -1, this is (really)
U 1199, 0C00,1002,403D,80C7,0131,E  ;28568      CLEAR FLAG0,NEXT/DS.DIVP.CC     ;this last Q high nibble, clear <C>
                                ;28569                                          ;image, do sign eval., write last byte
                                ;28570                                          ;&CC
                                ;28571
                                ;28572          ;11------------------------------;
                                ;28573          R[TEMP1]_(RB M[TEMP1]).RL.4,    ;Q(extra) < -1, get next [DV] digit
U 1198, 0C84,1237,0030,4047,0131,B  ;28574      NEXT/DS.DIVP.ML.SHIFT           ;from low nib. of TEMP1 into high nib.
                                ;28575                                          ;of low byte, enter [DV] shift code
```

```
                                ;28576          ;*****************************************************************************
                                ;28577          ; Old FLAG0=1 (TEMP0 is done), here on FLAG3? split, FLAG0 now has 0.
                                ;28578          ;*****************************************************************************
                                ;28579 =0
                                ;28580 DS.DIVP.ML.Q:
                                ;28581          ;0-------------------------------;
                                ;28582          M[TEMP4]_MB+1,SET FLAG3,        ;This byte has 1 digit (high byte),
                                ;28583          SIGND CMP?,NEXT/DS.DIVP.ML.Q1,  ;set flag for 2 digits/byte from
U 123C, 085E,4081,0B6D,8047,091A,0 380* ;28584  SIZE[LONG]                       ;now on, are we clear to write this
                                ;28585                                          ;byte?
                                ;28586
                                ;28587          ;1-------------------------------;
                                ;28588          M[TEMP4]_MB+1,SIZE[LONG],       ;This byte has 2 digits, do we clear
U 123D, 0886,4081,0B6D,8047,0919,C 380* ;28589  SIGND CMP?                       ;high nibble, not write, or ok?
                                ;28590
                                ;28591 =00
                                ;28592          ;00------------------------------;
                                ;28593          WRITE M[TEMP0],SIZE[BYTE],      ;>0, don't inc. it anymore, write
U 119C, 0C80,0592,4A40,05D8,0124,0      ;28594  WX.NE.0?,NEXT/DS.DIVP.ML.ZER    ;byte, is it =0?
                                ;28595
                                ;28596          ;01------------------------------;
                                ;28597          WB_M[TEMP0].AND.ZLIT0[0F0],     ;=0, don't inc. it anymore, high nibble
U 119D, 0180,0C12,0A37,8047,0123,E      ;28598  WX.EQ.0?,NEXT/DS.DIVP.ML.ZWR    ;must be zero (even dest.)
                                ;28599
                                ;28600          ;10------------------------------;
U 119E, 0986,4C11,0030,0847,0131,9      ;28601  M[TEMP4]_MB+ZLIT0[1]            ;<0, whole byte is N.G., inc.
                                ;28602                                          ;pointer for next time
                                ;28603 =
                                ;28604
                                ;28605          ;-------------------------------;
                                ;28606          WB_M[TEMP0],WX.NE.0?,           ;And check for lost data (ov)
U 1319, 0480,0592,4A70,0047,0124,2      ;28607  NEXT/DS.DIVP.ML.OVC
                                ;28608
                                ;28609          ;*****************************************************************************
                                ;28610          ; TEMP0 has full byte, FLAG3=1, and result of TEMP4_MB+1 was
                                ;28611          ; zero, here on WX.EQ.0? split.  If ov, we clear high nibble and
                                ;28612          ; set FLAG1.
                                ;28613          ;*****************************************************************************
                                ;28614 =0
                                ;28615 DS.DIVP.ML.ZWR:
                                ;28616          ;0-------------------------------;
                                ;28617          WRITE M[TEMP0].AND.ZLIT0[0F],   ;WRITE data with clear high nibble,
                                ;28618          SET FLAG1,WX.NE.0?,SIZE[BYTE],  ;set ov flag, split on zeroness of
U 123E, 0148,0C12,0A40,7DD8,0124,0      ;28619  NEXT/DS.DIVP.ML.ZER             ;written data
                                ;28620
                                ;28621          ;1-------------------------------;
                                ;28622          WRITE M[TEMP0],SIZE[BYTE],      ;WRITE data, split on zeroness
U 123F, 0C80,0592,4A40,05D8,0124,0      ;28623  WX.NE.0?,NEXT/DS.DIVP.ML.ZER
```

```
                                ;28624          ;*****************************************************************************
                                ;28625          ; Old FLAG0=1, Old FLAG3=0, here with 1 digit byte in TEMP0 on
                                ;28626          ; SIGND CMP? split from Q(extra) Q(extra)+1, FLAG0 has 0, FLAG3 has 1.
                                ;28627          ; If Q(extra) is <=0 we don't write byte, if >0 we do.
                                ;28628          ;*****************************************************************************
                                ;28629 =00
                                ;28630 DS.DIVP.ML.Q1:
                                ;28631          ;00-----------------------------;
                                ;28632          WRITE M[TEMP0],SIZE[BYTE],      ;>0, write byte, check for zeroness
U 11A0, 0C80,0592,4A40,05D8,0124,0 ;28633       WX.NE.0?,NEXT/DS.DIVP.ML.ZER
                                ;28634
                                ;28635          ;01-----------------------------;
                                ;28636          WB_M[TEMP0],WX.NE.0?,           ;=0, if data>0 we have ov condition
U 11A1, 0480,0592,4A70,0047,0124,2 ;28637       NEXT/DS.DIVP.ML.OVC
                                ;28638
                                ;28639          ;10-----------------------------;
                                ;28640          WB_M[TEMP0],WX.NE.0?,           ;<0, if data>0 we have ov condition
U 11A2, 0480,0592,4A70,0047,0124,2 ;28641       NEXT/DS.DIVP.ML.OVC
                                ;28642 =
                                ;28643
                                ;28644
                                ;28645          ;*****************************************************************************
                                ;28646          ; Here on:
                                ;28647          ; 1) Old FLAG0=1, FLAG3=1, Q(extra)>0: full byte, written to dest.,
                                ;28648          ;                FLAG0 now has 0.
                                ;28649          ; 2) Old FLAG0=1, FLAG3=1, Q(extra)=0 on first inc.: high nibble was
                                ;28650          ;                zeroed before writing if it was nonzero, FLAG0 has 0.
                                ;28651          ; 3) Old FLAG0=1, Old FLAG3=0, Q(extra)>=0: 1 digit in byte, FLAG0 now
                                ;28652          ;                has 0, FLAG3 now has 1.
                                ;28653          ;
                                ;28654          ; Byte was written to destination, and branch is taken to here on
                                ;28655          ; WX.NE.0? split from byte data.
                                ;28656          ;*****************************************************************************
                                ;28657 =0
                                ;28658 DS.DIVP.ML.ZER:
                                ;28659          ;0-----------------------------;
                                ;28660          VA_M[VA]+ZLITO[1],              ;Data=0
U 1240, 0D81,BC11,0030,0CA7,0131,A ;28661       NEXT/DS.DIVP.ML.ZER.1
                                ;28662
                                ;28663          ;1-----------------------------;
                                ;28664          CLEAR FLAG2,                    ;Data>0
U 1241, 0D11,BC11,0030,0CA7,0131,A ;28665       VA_M[VA]+ZLITO[1]
                                ;28666
                                ;28667 DS.DIVP.ML.ZER.1:
                                ;28668          ;-----------------------------;
                                ;28669          M[TEMP5]_MB-ZLITO[1],           ;Dec. Q(LB)
U 131A, 0986,5C10,0030,0847,0124,2 ;28670       NEXT/DS.DIVP.ML.OVC
```

```
                                ;28671          ;*********************************************************************
                                ;28672          ; Here on:
                                ;28673          ; 1) From DS.DIVP.ML.ZER: code (all conditions listed in DS.DIVP.ML.ZER
                                ;28674          ;                  banner, after <Z> bit mask and VA increment) directly.
                                ;28675          ; 2) Old FLAG0=1, FLAG3=1, Q(extra)<0: Whole byte was N.G., WX.NE.0?
                                ;28676          ;                  split taken here, FLAG0 now is 0.
                                ;28677          ; 3) Old FLAG0=1, Old FLAG3=0, Q(extra)<0: Ditto, FLAG0 has 0,
                                ;28678          ;                  FLAG3 has 1.
                                ;28679          ;
                                ;28680          ; Except for case 1, =0 is data was =0, =1 is data >0, we have ov
                                ;28681          ; condition.
                                ;28682          ;*********************************************************************
                                ;28683 =0
                                ;28684 DS.DIVP.ML.OVC:
                                ;28685          ;0-------------------------------;
                                ;28686          R[TEMP1]_ZEXT(XB) PC_PC+1,      ;Data=0, get next byte
U 1242, 0085,759E,4000,6047,0131,B ;28687       NEXT/DS.DIVP.ML.SHIFT
                                ;28688
                                ;28689          ;1-------------------------------;
                                ;28690          SET FLAG1,                      ;Data>0, ov condition, set <V> image,
U 1243, 084D,759E,4000,6047,0131,B ;28691       R[TEMP1]_ZEXT(XB) PC_PC+1       ;ditto
                                ;28692
                                ;28693
                                ;28694          ;*********************************************************************
                                ;28695          ; TEMP1 has either next byte from DV or last byte from DV with the
                                ;28696          ; low nibble shifted into the high nibble position ( low byte ).
                                ;28697          ; We get the next digit to shift into [DV] into the highest nibble
                                ;28698          ; position of TEMP7 and begin the shift.
                                ;28699          ;*********************************************************************
                                ;28700
                                ;28701 DS.DIVP.ML.SHIFT:
                                ;28702          ;-------------------------------;
U 131B, 0C86,72B7,0000,4047,011A,4 ;28703       M[TEMP7]_R[TEMP1].RR.8          ;TEMP7<31:28> has next digit
```

```
                                ;28704 .TOC   "      Decimal String                  :      Shift in next [DV] digit"
                                ;28705
                                ;28706          ;*************************************************************************
                                ;28707          ; All writing, ov checking and zero dest. checking have been done.
                                ;28708          ; TEMP7 high nibble, byte#3 has [DV] digit to shift into LSD position
                                ;28709          ; in MTEMPs.
                                ;28710          ;*************************************************************************
                                ;28711 =00
                                ;28712          ;00-----------------------------;
                                ;28713          PUSH,R[TEMP3]_M[TEMP8].BCDSWP,  ;TEMP3_old MTEMP8, Q_new MTEMP8
U 11A4, 0434,8637,0030,C047,053E,B ;28714       NEXT/DS.SR.25
                                ;28715
                                ;28716          ;01-----------------------------;
                                ;28717          PUSH,M[TEMP8]_Q,                ;MTEMP8_new MTEMP8, BCDSWP it
U 11A5, 0086,803A,403D,8047,053E,8 ;28718       NEXT/DS.SR.22
                                ;28719
                                ;28720          ;10-----------------------------;
U 11A6, 0484,9637,0031,C047,0131,C ;28721       R[TEMP7]_M[TEMP9].BCDSWP        ;TEMP7_old MTEMP9
                                ;28722 =
                                ;28723
                                ;28724          ;-------------------------------;
U 131C, 0880,7237,1030,C047,011A,8 ;28725       Q_(R[TEMP3] M[TEMP7]).RL.4      ;Q_new MTEMP9
                                ;28726
                                ;28727 =00
                                ;28728          ;00-----------------------------;
                                ;28729          PUSH,M[TEMP9]_Q,                ;MTEMP9_new MTEMP9, BCDSWP it
U 11A8, 0C86,903A,403D,8047,053E,9 ;28730       NEXT/DS.SR.23
                                ;28731
                                ;28732          ;01-----------------------------;
                                ;28733          PUSH,R[TEMP3]_M[TEMP10].BCDSWP, ;TEMP3_old MTEMP10, Q_new MTEMP10
U 11A9, 0084,A637,0030,C047,053E,B ;28734       NEXT/DS.SR.25
                                ;28735
                                ;28736          ;10-----------------------------;
                                ;28737          PUSH,M[TEMP10]_Q,               ;MTEMP10_new MTEMP10, BCDSWP it
U 11AA, 0C86,A03A,403D,8047,053E,A ;28738       NEXT/DS.SR.24
                                ;28739
                                ;28740          ;11-----------------------------;
                                ;28741          M[ERRCOD]_MB.BCDSWP,            ;MERRCOD_old MERRCOD (BCDSWaPed)
U 11AB, 08B6,B637,0030,0047,0131,D ;28742       STEPC_14.                       ;Load stepc for aborting main loop.
                                ;28743
                                ;28744          ;-------------------------------;
                                ;28745          M[ERRCOD]_(R[TEMP3] MB).RL.4,   ;MERRCOD_new MERRCOD
U 131D, 009E,B237,0B30,D8E7,0124,4 ;28746       DEC STEPC,IP.TS?                ;STEPC_13
                                ;28747
                                ;28748 =0       ;0-----------------------------;STEPC_12
                                ;28749          M[ERRCOD]_MB.BCDSWP,DEC STEPC,  ;No ints. pending, all done, loop back
U 1244, 0C9E,B637,0430,0047,0119,4 ;28750       FLAG0?,NEXT/DS.DIVP.ML.T0       ;for next Q byte, zero TEMP0 if last
                                ;28751                                          ;TEMPC was a finished Q byte
                                ;28752
                                ;28753          ;1-----------------------------;
                                ;28754          PUSH,INTPEND OR TIMER?,         ;Something pending, return -1 or
U 1245, 0480,0036,4AF0,0047,04F7,0 ;28755       NEXT/IE.SERV.IP.TS2             ;branch to IE.PACK.DONE
```

```
                                ;28756 .TOC     ''      Decimal String                  :       Setting CC''
                                ;28757
                                ;28758          ;*************************************************************************
                                ;28759          ; FLAG0 (<C> bit image) is zero, ALUS has BCDSIGN data
                                ;28760          ;*************************************************************************
                                ;28761
                                ;28762 DS.DIVP.CC:
                                ;28763          ;-------------------------------;
                                ;28764          WB_M[TEMP0],CLEAR FLAG3,        ;Clear <N> bit image (guess pos.),
U 131E, 0418,0592,4A70,0047,0124,6 ;28765       WX.NE.0?                        ;is data=0?
                                ;28766
                                ;28767 =0
                                ;28768          ;0------------------------------;
                                ;28769          M[TEMP0]_ZLIT0[0C],             ;Yes, dest. is +0 guess
U 1246, 0186,0C37,0030,6047,0131,F ;28770       NEXT/DS.DIVP.CC.DVSGN
                                ;28771
                                ;28772          ;1------------------------------;
                                ;28773          WB_M[TEMP4]+ZLIT0[1],           ;No, do we zero high nibble?
U 1247, 0580,4C11,0A70,0847,0924,8 378* ;28774  WX.NE.0?
                                ;28775
                                ;28776 =0
                                ;28777          ;0------------------------------;
                                ;28778          SET FLAG1,M[TEMP0]_ZLIT0[0C],   ;Yes, we have ov (<V>_1), dest._+0
U 1248, 094E,0C37,0030,6047,0131,F ;28779       NEXT/DS.DIVP.CC.DVSGN           ;guess
                                ;28780
                                ;28781          ;1------------------------------;
                                ;28782          CLEAR FLAG2,                    ;No, dest._+(nonzero), <Z> image_0
U 1249, 0116,0C12,4030,6047,0131,F ;28783       M[TEMP0]_MB.OR.ZLIT0[0C]
                                ;28784
                                ;28785
                                ;28786 DS.DIVP.CC.DVSGN:
                                ;28787          ;-------------------------------;
U 131F, 0C80,0036,4B70,1847,011A,D ;28788       ALUS?                           ;What is DV sign?
                                ;28789
                                ;28790 =01
                                ;28791          ;01-----------------------------;
                                ;28792          WB_M[TEMP2].AND.ZLIT0[01],      ;Positive, what is DR sign?
U 11AD, 0580,2C12,0A70,0847,0124,A ;28793       WX.NE.0?,NEXT/DS.DIVP.CC.DRSGN
                                ;28794
                                ;28795          ;11-----------------------------;
                                ;28796          WB_M[TEMP2].AND.ZLIT0[01],      ;Negative, ditto
U 11AF, 0180,2C12,0A30,0847,0124,A ;28797       WX.EQ.0?
```

```
                                ;28798          ;*************************************************************************
                                ;28799          ; =0 is (DV+, DR-) OR (DV-, DR+): Q is negative (we think).
                                ;28800          ; =1 is (DV+, DR+) OR (DV-, DR-): Q is positive.
                                ;28801          ;*************************************************************************
                                ;28802 =0
                                ;28803 DS.DIVP.CC.DRSGN:
                                ;28804          ;0-----------------------------;
                                ;28805          SET FLAG3,FLAG<2-0>?,           ;Q may be negative, set <N> image,
U 124A, 0858,0036,45B0,0047,0104,8  ;28806      NEXT/DS.DIVP.CC.QNP             ;split on <Z> and <V> images
                                ;28807
                                ;28808          ;1-----------------------------;
                                ;28809          WRITE M[TEMP0],SIZE[BYTE],      ;Q is positive, write byte as is, wrap
U 124B, 0080,0592,4000,05D8,0132,0  ;28810      NEXT/DS.DIVP.CC.EXIT            ;things up
                                ;28811
                                ;28812
                                ;28813
                                ;28814          ;*************************************************************************
                                ;28815          ; Here from DS.DIVP.CC.DRSGN on FLAG<2-0>? BUT.  =00* is Q>0, no ov,
                                ;28816          ; dest. is neg, CC is neg.  =01* is Q>0, ov, dest. is neg, CC is neg.
                                ;28817          ; =10* is Q=0, no ov, dest. is pos zero, CC is pos zero.  =11* is
                                ;28818          ; Q=0, ov, dest. is neg zero, CC is pos zero.
                                ;28819          ;*************************************************************************
                                ;28820 =00*
                                ;28821 DS.DIVP.CC.QNP:
                                ;28822          ;00*----------------------------;
                                ;28823          WRITE M[TEMP0].OR.ZLIT0[0D],    ;Negative dest.
U 1048, 0580,0C12,4000,6DD8,0132,0  ;28824      SIZE[BYTE],NEXT/DS.DIVP.CC.EXIT
                                ;28825
                                ;28826          ;01*----------------------------;
                                ;28827          WRITE M[TEMP0].OR.ZLIT0[0D],    ;Negative dest.
U 104A, 0580,0C12,4000,6DD8,0132,0  ;28828      SIZE[BYTE],NEXT/DS.DIVP.CC.EXIT
                                ;28829
                                ;28830          ;10*----------------------------;
                                ;28831          WRITE M[TEMP0],SIZE[BYTE],      ;Pos. dest., pos. CC (clear <N>)
U 104C, 0818,0592,4000,05D8,0132,0  ;28832      CLEAR FLAG3,NEXT/DS.DIVP.CC.EXIT
                                ;28833
                                ;28834          ;11*----------------------------;
                                ;28835          WRITE M[TEMP0].OR.ZLIT0[0D],    ;Neg. dest., pos. CC (clear <N>)
U 104E, 0D18,0C12,4000,6DD8,0132,0  ;28836      SIZE[BYTE],CLEAR FLAG3
```

```
                                        ;28837          ;*************************************************************************
                                        ;28838          ; FLAGS has CCimage, dest. is written.  Fix stack pointer & PC and
                                        ;28839          ; finish inst. thru ASHP code.  DS.DIVP.BAD: is to restore PC after
                                        ;28840          ; an invalid DV string or an invalid interrupt code encountered during
                                        ;28841          ; unpacking.
                                        ;28842          ;*************************************************************************
                                        ;28843
                                        ;28844 DS.DIVP.CC.EXIT:
                                        ;28845          ;-------------------------------;
U 1320, 0C87,929C,7024,0047,0132,1      ;28846          MTEMP0_R[R0].RR.24 Q_PCBACK     ;TEMP0_deltaPC, Q_PCBACK
                                        ;28847
                                        ;28848          ;-------------------------------;
U 1321, 0480,0019,0000,0487,0132,2      ;28849          PC_ZEXT(M[TEMP0])+Q,SIZE[BYTE]  ;PC is fixed
                                        ;28850
                                        ;28851          ;-------------------------------;
U 1322, 0484,05B7,0035,0047,0132,3      ;28852          R[R4]_0                         ;Fix 1 GPR
                                        ;28853
                                        ;28854          ;-------------------------------;
U 1323, 0186,0C37,0030,8047,0132,4      ;28855          M[TEMP0]_ZLIT0[16.]             ;Stack workspace area
                                        ;28856
                                        ;28857          ;-------------------------------;
                                        ;28858          R[SP]_M[TEMP0]+RB,              ;(SP) has old (before DIVP) value,
U 1324, 0884,0001,0037,8047,0110,8      ;28859          NEXT/DS.ASHP.FLAGS              ;set CC & finish
                                        ;28860
                                        ;28861
                                        ;28862
                                        ;28863 DS.DIVP.BAD:
                                        ;28864          ;-------------------------------;
                                        ;28865          PC_ZEXT(M[TEMP0])+Q,SIZE[BYTE], ;IRD1 in CMPP3 code
U 1325, 0480,0019,0000,0487,0102,7      ;28866          NEXT/DS.CMPP.EXIT
```

```
;28867 .TOC     "       Decimal String                  :       DIVP Pack routine"
;28868
;28869          ;*************************************************************************
;28870          ; There are 5 DIVP interrupt codes, 0-4.  Code 0 is a StateSave which
;28871          ; was made before entering the main division, no packing is done.
;28872          ; Codes 1-4 enter this code in a variety of ways:
;28873          ; Codes 1 & 2: enter at DS.DIVP.PCK.12.CON: with the int. code
;28874          ; masked in GPR0.
;28875          ; Code 3: enters at DS.DIVP.PCK.34: with TEMP6 already masked into
;28876          ;       GPR4 and the interrupt code in TEMP3.
;28877          ; Code 4: enters at DS.DIVP.PCK.4 with the interrupt code in TEMP3.
;28878          ;               * appended to input values means >= zero,
;28879          ;               less than 1 byte in length
;28880          ;
;28881          ;       Input           FLAGS*          Current FLAG status (#4)
;28882          ;                       TEMP1*          Last read DV byte (#3,4)
;28883          ;                       TEMP2*          DR sign value (#3,4)
;28884          ;                       TEMP3*          Int. code (#1,2,3,4)
;28885          ;                       TEMP4           DR(LD) (#1)
;28886          ;                                       Garbage (#2)
;28887          ;                                       DV(LDNLZ) (#3)
;28888          ;                                       Q(extra) (#4)
;28889          ;                       TEMP5*          DV(LD) (#1,2,3,4)
;28890          ;                       TEMP6*          Q(LD)/2 (#1,2,3,4)
;28891          ;                       GPR0*           (PC-PCBACK) in byte #3
;28892          ;                                       (#1,2,3,4)
;28893          ;                       GPR2*           (DR(LDNLZ)) in byte #0
;28894          ;                                       (#2,3,4)
;28895          ;                                       0 (#1)
;28896          ;                       GPR3            DV base address (#1,2,3,4)
;28897          ;                       GPR4            Untouched (#0)
;28898          ;                                       (DR(LD)) in byte #2, zero
;28899          ;                                       in other 3 bytes (#1,2,3,4)
;28900          ;                       GPR5            Q base address (#1,2,3,4)
;28901          ;                       PC              Pointer into DV string
;28902          ;                                       (#3,4)
;28903          ;                       VA              Pointer into Q string
;28904          ;                                       (#4)
;28905          ;
;28906          ;       Resources       TEMP0           Misc. data holding
```

```
                                ;28907          ;       Output          <All int. codes output the followirg>
                                ;28908          ;                       GPR0    #2      Int. code
                                ;28909          ;                               #3      (PC-PCBACK)
                                ;28910          ;                       GPR2    #1      TEMP5
                                ;28911          ;                               #2      TEMP4
                                ;28912          ;                       GPR4    #2      (DR(LD))
                                ;28913          ;                       <Additional packs:>
                                ;28914          ;               Code #1 GPR4    #3      TEMP6
                                ;28915          ;               Code #2 GPR2    #0      (DR(LDNLZ)) on input to pack
                                ;28916          ;                                       routine
                                ;28917          ;                       GPR4    #3      TEMP6
                                ;28918          ;               Code #3 GPR0    #1      TEMP2
                                ;28919          ;                       GPR2    #0      (DR(LDNLZ)) on input to pack
                                ;28920          ;                                       routine
                                ;28921          ;                               #3      TEMP1
                                ;28922          ;                       GPR4    #1      PC(XB)-GPR3  (>= zero)
                                ;28923          ;                               #3      TEMP6
                                ;28924          ;               Code #4 GPR0    #0      FLAGS
                                ;28925          ;                               #1      TEMP2
                                ;28926          ;                       GPR2    #0      (DR(LDNLZ)) on input to pack
                                ;28927          ;                                       routine
                                ;28928          ;                               #3      TEMP1
                                ;28929          ;                       GPR4    #0      VA-GPR5  (>= zero)
                                ;28930          ;                               #1      PC(XB)-GPR3  (>= zero)
                                ;28931          ;                               #3      TEMP6
                                ;28932          ;*******************************************************************************
                                ;28933
                                ;28934
                                ;28935          ;*******************************************************************************
                                ;28936          ; Code 1 and 2 packs enter here
                                ;28937          ;*******************************************************************************
                                ;28938 =0
                                ;28939 DS.DIVP.PCK.12.CON:
                                ;28940          ;0-----------------------------;
                                ;28941          PUSH,NEXT/DS.SR.28,             ;Pack into GPR2
J 124C, 0084,53BE,4024,8047,053E,6      ;28942          R[R2]_RB.OR.(M[TEMP5].RR.24)
                                ;28943
                                ;28944          ;1-----------------------------;
                                ;28945          R[R4]_RB.OR.(M[TEMP6].RR.8),    ;Load GPR4, service int.
J 124D, 0084,53BE,4005,0047,00FE,2      ;28946          NEXT/IE.PACK.DONE
```

```
                                        ;28947          ;*****************************************************************
                                        ;28948          ; Code 4 pack
                                        ;28949          ;*****************************************************************
                                        ;28950
                                        ;28951 DS.DIVP.PCK.4:
                                        ;28952          ;-------------------------------;
U 1326, 0C87,B000,0035,4047,0132,7      ;28953          MTEMP0_VA-R[R5]                 ;Calc. VA offset
                                        ;28954
                                        ;28955          ;-------------------------------;
U 1327, 0C84,0002,4035,0047,0132,8      ;28956          R[R4]_M[TEMP0].OR.RB            ;Masked into GPR4
                                        ;28957
                                        ;28958          ;-------------------------------;
U 1328, 0482,0036,4004,0387,0132,9      ;28959          R[R0].SIZ_FLAGS,SIZE[BYTE]      ;FLAGS
                                        ;28960
                                        ;28961          ;-------------------------------;
U 1329, 0084,63BE,4005,0047,0132,A      ;28962          R[R4]_RB.OR.(M[TEMP6].RR.8)     ;TEMP6, enter common pack with code 3
                                        ;28963
                                        ;28964
                                        ;28965          ;*****************************************************************
                                        ;28966          ; Code 3 pack and continuation of code 4 pack
                                        ;28967          ;*****************************************************************
                                        ;28968
                                        ;28969 DS.DIVP.PCK.34:
                                        ;28970          ;-------------------------------;
U 132A, 0884,33BE,4014,0047,0132,B      ;28971          R[R0]_RB.OR.(M[TEMP3].RR.16)    ;Load int. code
                                        ;28972
                                        ;28973          ;-------------------------------;
U 132B, 0087,A000,0034,C047,0124,E      ;28974          MTEMP0_PC-R[R3]                 ;Make PC offset
                                        ;28975
                                        ;28976 =0
                                        ;28977          ;0-----------------------------;
                                        ;28978          R[R4]_RB.OR.(M[TEMP0].RR.24),   ;Mask in PC offset, load GPR2
U 124E, 0484,03BE,4025,0047,053E,4      ;28979          PUSH,NEXT/DS.SR.21
                                        ;28980
                                        ;28981          ;-------------------------------;
                                        ;28982          R[R0]_RB.OR.(M[TEMP2].RR.24),   ;All done
U 124F, 0484,23BE,4024,0047,00FE,2      ;28983          NEXT/IE.PACK.DONE
```

```
;28984 .TOC     "       Decimal String                  :       DIVP Unpack routine"
;28985
;28986          ;*************************************************************************
;28987          ; There are 5 DIVP interrupt codes, 0-4.
;28988          ; Code 0 is the StateSave made before entering DIVP main loops.
;28989          ; Code 1 is int. check before reading DR during initialization or
;28990          ;       BUS exception/int. pending in DS.READ during init.
;28991          ; Code 2 is any int. pending or BUS exception during [DV] read during
;28992          ;       initialization, up to but not including the int. check made
;28993          ;       before the zero-high-Q-digits code.
;28994          ; Code 3 is the int. check made before deciding whether to zero high
;28995          ;       Q digits during init.
;28996          ; Code 4 is int. pending or BUS exception during the zeroing of high
;28997          ;       Q digits.
;28998          ;
;28999          ; Illegal values in the GPR's are corrected during unpack except for an
;29000          ; illegal interrupt code, which causes branching to
;29001          ; DS.CCPCIRD1.DIVP.BAD.
;29002          ;
;29003          ; See the banner page for DIVP Pack routine for a listing of all
;29004          ; values saved for the different interrupt codes.
;29005          ;
;29006          ;       Input           <From initialization:>
;29007          ;                       GPR1            Base address, DR
;29008          ;                       GPR3            Base address, DV
;29009          ;                       GPR5            Base address, Q
;29010          ;                       <All int. codes pack the following:>
;29011          ;                       GPR0    #2      Int. code
;29012          ;                               #3      (PC-PCBACK)
;29013          ;                       GPR2    #1      TEMP5
;29014          ;                               #2      TEMP4
;29015          ;                       GPR4    #2      (DR(LD))
;29016          ;                       <Additional packs:>
;29017          ;               Code #1 GPR4    #3      TEMP6
;29018          ;               Code #2 GPR2    #0      (DR(LDNLZ))
;29019          ;                       GPR4    #3      TEMP6
;29020          ;               Code #3 GPR0    #1      TEMP2
;29021          ;                       GPR2    #0      (DR(LDNLZ))
;29022          ;                               #3      TEMP1
;29023          ;                       GPR4    #1      PC(XB)-GPR3  (>= zero)
;29024          ;                               #3      TEMP6
;29025          ;               Code #4 GPR0    #0      FLAGS
;29026          ;                               #1      TEMP2
;29027          ;                       GPR2    #0      (DR(LDNLZ))
;29028          ;                               #3      TEMP1
;29029          ;                       GPR4    #0      VA-GPR5  (>= zero)
;29030          ;                               #1      PC(XB)-GPR3  (>= zero)
;29031          ;                               #3      TEMP6
;29032          ;
;29033          ;       Resources       QREG            Misc. temp.
;29034          ;                       DREG            Ditto
;29035          ;                       TEMP0           Misc. data, PC restore
;29036          ;                       TEMP3           Misc. data
```

```
;29037          ;
;29038          ;       Output          FLAGS           FLAGS (#0,3,4)
;29039          ;                                       (only necessary for #0 & 4)
;29040          ;                       TEMP1           Last read DV byte (#0,3,4)
;29041          ;                       TEMP2           DR sign value (#0,3,4)
;29042          ;                       TEMP4           DR(LD) (#1)
;29043          ;                                       Garbage (#2)
;29044          ;                                       DV(LDNLZ) (#3)
;29045          ;                                       Q(extra) (#0,4)
;29046          ;                       TEMP5           DV(LD) (#1,2,3,4)
;29047          ;                                       Q(LB)-1 (#0)
;29048          ;                       TEMP6           Q(LD)/2 (#1,2,3,4)
;29049          ;                       RTEMP8-RTEMP11  DR restored by calling
;29050          ;                                       DS.READ (#0,2,3,4)
;29051          ;                       MTEMP8-MERRCOD  [DV] restored from stack
;29052          ;                                       (#0,3,4))
;29053          ;                                       *MERRCOD is not needed after any
;29054          ;                                       BUS exception*
;29055          ;                       GPR0            (PC-PCBACK) in byte #3, low
;29056          ;                                       3 bytes zero (#1,2,3,4)
;29057          ;                                       Untouched (#0)
;29058          ;                       GPR2            (DR(LDNLZ)) in byte #0, high
;29059          ;                                       3 bytes zero (#1,2,3,4)
;29060          ;                                       Untouched (#0)
;29061          ;                       GPR4            Untouched (#0)
;29062          ;                                       (DR(LD)) in byte #2, zero filled
;29063          ;                                       in other 3 bytes (#1,2,3,4)
;29064          ;                       PC              Pointer into DV string
;29065          ;                                       (#0,3,4)
;29066          ;                                       DV Base address (#2)
;29067          ;                       VA              Pointer into Q string
;29068          ;                                       (#0,4)
;29069          ;                       ALUS            DR sign evaluation data (#2)
               ;*******************************************************************************
```

```
                                         ;29070 .REGION/IRD1.R1L,IRD1.R1H
                                         ;29071 =000
                                         ;29072 DS.DIVP.UNP:
                                         ;29073         ;000-----------------------------;
U 0350, 0D86,CC37,0030,0047,0132,C       ;29074         M[FPDOFFSET]_ZLITO[0]            ;Zero FPDOFFSET
                                         ;29075 =
                                         ;29076
                                         ;29077 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                         ;29078
                                         ;29079         ;--------------------------------;
U 132C, 0B80,0707,0404,0047,0132,D       ;29080         LONLIT_[01F1F7F7F]               ;Ensure GPR4 has correct values
                                         ;29081
                                         ;29082         ;--------------------------------;
U 132D, 0C86,05BE,403D,4047,0132,E       ;29083         M[TEMP0]_R[LONLIT]
                                         ;29084
                                         ;29085         ;--------------------------------;
U 132E, 0084,0002,0035,0047,0132,F       ;29086         R[R4]_M[TEMP0].AND.RB
                                         ;29087
                                         ;29088         ;--------------------------------;
U 132F, 0D86,0C37,0030,F847,0133,0       ;29089         M[TEMP0]_ZLITO[1F]               ;Ensure DR(LDNLZ) is legal
                                         ;29090
                                         ;29091         ;--------------------------------;
                                         ;29092         R[R2].SIZ_M[TEMP0].AND.RB,       ;Proper value is 1-1F, is it zero?
U 1330, 0482,0002,0A44,8047,0125,0       ;29093         SIZE[BYTE],WX.NE.0?
                                         ;29094
                                         ;29095 =0
                                         ;29096         ;0-------------------------------;
U 1250, 0884,073E,4004,8047,0125,1       ;29097         R[R2]_RB.OR.CONX(1)              ;Yes, make it 1
                                         ;29098
                                         ;29099         ;1-------------------------------;
U 1251, 0486,42B7,0014,8047,0133,1       ;29100         M[TEMP4]_R[R2].RR.16             ;Load TEMP4 data
                                         ;29101
                                         ;29102         ;--------------------------------;
U 1331, 0486,52B7,0004,8047,0133,2       ;29103         M[TEMP5]_R[R2].RR.8              ;Load TEMP5 data
                                         ;29104
                                         ;29105         ;--------------------------------;
U 1332, 0C86,65BE,4034,8047,0133,3       ;29106         M[TEMP6]_R[R2]                   ;Save GPR2 incase of packup during
                                         ;29107                                          ;call to DS.READ
```

```
                                ;29108          ;-------------------------------;
U 1333, 0486,02B7,0014,0047,0133,4  ;29109      M[TEMP0]_R[R0].RR.16           ;Int. code byte to byte#0
                                ;29110
                                ;29111          ;-------------------------------;
U 1334, 0986,0C12,0030,3847,0133,5  ;29112      M[TEMP0]_MB.AND.ZLIT0[07]      ;Mask off unwanted bits
                                ;29113
                                ;29114          ;-------------------------------;
                                ;29115          WB_M[TEMP0]-ZLIT0[5],          ;Is int. code 5, 6, or 7?
U 1335, 0980,0C10,06F0,2847,091B,1 377*  ;29116 WB<31-30>?
                                ;29117
                                ;29118 =01
                                ;29119          ;01-----------------------------;
                                ;29120          MTEMP0_R[R0].RR.24 Q_PCBACK,   ;Yes, illegal FPD situation,
U 11B1, 04E7,929C,7024,0047,0132,5  ;29121      NEXT/DS.DIVP.BAD,CLEAR FPD     ;terminate bad inst. now
                                ;29122
                                ;29123          ;11-----------------------------;
U 11B3, 0480,0592,4230,0047,010C,8  ;29124      WB_M[TEMP0],WB<5-0>?           ;No, it's 0-4, what is it?
                                ;29125
                                ;29126 10C8:
                                ;29127 ;=000                                   ;CAN'T FORCE AN ADDRESS AND CONSTRAIN
                                ;29128          ;000----------------------------;
                                ;29129          M[TEMP4]_SEXT(MB),SIZE[BYTE],  ;StateSave, restore TEMP4
U 10C8, 0886,4816,400D,8047,0125,2  ;29130      NEXT/DS.DIVP.UNP.0
                                ;29131
                                ;29132 10C9:    ;001----------------------------;
                                ;29133          M[TEMP4]_MB.AND.ZLIT0[1F],     ;Code 1, restore TEMP4 & ensure legal
U 10C9, 0586,4C12,0030,F847,0133,8  ;29134      NEXT/DS.DIVP.UNP.1             ;value
                                ;29135
                                ;29136 10CA:    ;010----------------------------;
                                ;29137          M[TEMP5]_MB.AND.ZLIT0[1F],     ;Code 2, TEMP4 is not needed, fix
U 10CA, 0186,5C12,0030,F847,0533,A  ;29138      PUSH,NEXT/DS.DIVP.UNP.2        ;TEMP5, RETURN [+6]
                                ;29139
                                ;29140 10CB:    ;011----------------------------;
                                ;29141          M[TEMP4]_MB.AND.ZLIT0[1F],     ;Code 3, ditto, 0 is illegal TEMP4
U 10CB, 0186,4C12,0A70,F847,011B,4  ;29142      WX.NE.0?,NEXT/DS.DIVP.UNP.3    ;value, is it nonzero?
                                ;29143
                                ;29144 10CC:    ;100----------------------------;
                                ;29145          M[TEMP4]_SEXT(MB),SIZE[BYTE],  ;Code 4, ditto, anything is legal
U 10CC, 0886,4816,400D,8047,0125,8  ;29146      NEXT/DS.DIVP.UNP.4
                                ;29147 =
```

```
                                        ;29148          ;*****************************************************************************
                                        ;29149          ; Continuation of code 0 (StateSave) unpack
                                        ;29150          ;*****************************************************************************
                                        ;29151 =0
                                        ;29152 DS.DIVP.UNP.0:
                                        ;29153          ;0-------------------------------;
                                        ;29154          M[TEMP5]_ZEXT(MB),PUSH,         ;Restore TEMP5, restore DR, [DV], and
                                        ;29155          SIZE[BYTE],                     ;PC
U 1252, 0086,559E,4000,0047,0533,C      ;29156          NEXT/DS.DIVP.UNP.C0.034
                                        ;29157
                                        ;29158          ;1-------------------------------;
U 1253, 0C86,05BE,4035,0047,0133,6      ;29159          M[TEMP0]_R[R4]                  ;Restore VA
                                        ;29160
                                        ;29161          ;-------------------------------;
                                        ;29162          VA_ZEXT(M[TEMP0])+R[R5],
U 1336, 0080,0015,0005,44A7,0133,7      ;29163          SIZE[BYTE]
                                        ;29164
                                        ;29165          ;-------------------------------;
                                        ;29166          R[R2]_M[TEMP6],                 ;Restore GPR2 to StateSave value,
U 1337, 0C84,6592,4034,8047,0131,1      ;29167          NEXT/DS.DIVP.REE.0              ;reenter main loop after FPDOFFSET_11.
                                        ;29168
                                        ;29169
                                        ;29170          ;*****************************************************************************
                                        ;29171          ; Continuation of code 1 unpack
                                        ;29172          ;*****************************************************************************
                                        ;29173
                                        ;29174 DS.DIVP.UNP.1:
                                        ;29175          ;-------------------------------;
U 1338, 0986,5C12,0030,F847,0125,4      ;29176          M[TEMP5]_MB.AND.ZLIT0[1F]       ;TEMP5 must be 0-1F
                                        ;29177
                                        ;29178 =0
                                        ;29179          ;0-------------------------------;
                                        ;29180          M[TEMP6]_R[R4].RR.24,            ;Get TEMP6 data into low byte, fix
U 1254, 0886,62B7,0025,0047,053D,F      ;29181          PUSH,NEXT/DS.SR.26              ;GPR0, GPR2, and GPR4
                                        ;29182
                                        ;29183          ;1-------------------------------;
                                        ;29184          M[TEMP6]_ZEXT(MB),SIZE[BYTE],   ;Fix TEMP6, all done
U 1255, 0086,659E,4000,0047,011F,C      ;29185          NEXT/DS.DIVP.REE.1
```

```
                                ;29186          ;*****************************************************************************
                                ;29187          ; Continuation of code 2 unpack
                                ;29188          ;*****************************************************************************
                                ;29189
                                ;29190 ;DS.DIVP.UNP.2:
                                ;29191 ;        ;--------------------------------;
                                ;29192 ;        M[TEMP0]_R[R4].RR.16             ;Byte#0_DR(LD)
                                ;29193 ;
                                ;29194 ;        ;--------------------------------;
                                ;29195 ;        STEPC_D_SEXT(M[TEMP0]).SR.1,     ;STEPC_D_L/2 for DR read
                                ;29196 ;        SIZE[BYTE]
                                ;29197
                                ;29198 1339:
                                ;29199 MM.PB.WRITE5:
                                ;29200          ;--------------------------------;
                                ;29201          FLAGS_R[RTMPGPR],                ; RESTORE FLAGS
U 1339, 0080,05BE,4037,C307,0176,7  ;29202      NEXT/MM.PB.WRITE10               ; See memory management module
                                ;29203
                                ;29204 133D:
                                ;29205 MM.PB.PROBE:                              ; Hope for location 104A
                                ;29206          ;--------------------------------;
                                ;29207          PROBE WRITE?,SIZE[BYTE],         ; PROBE VA USING CURRENT MODE.
                                ;29208          R[RTMPGPR]_FLAGS,                ; SAVE THE CURRENT EVENT FLAGS
U 133D, 0484,0036,4787,C39D,0961,C 479*  ;29209 NEXT/OLD16TC                     ; MAKE SURE WE SKIP OLD 1769
                                ;29210
                                ;29211 ;=000
                                ;29212 10D0:    ;000-----------------------------;
                                ;29213          PUSH,VA_D+R[R1]+1,               ;Point VA past sign byte of DR, read
U 10D0, 0480,00A1,0034,44A7,053C,0  ;29214      NEXT/DS.READ.DIVP.DR             ;DR
                                ;29215
                                ;29216 ;=010
                                ;29217 10D2:    ;010-----------------------------;
                                ;29218          R[R2]_M[TEMP6],          ;BUS exception or int. pending during
U 10D2, 0484,6592,4034,8047,00FE,2  ;29219      NEXT/IE.PACK.DONE                ;DS.READ, GPR's are untouched,
                                ;29220                                           ;reload GPR2 & directly to IANDE
                                ;29221
                                ;29222 10D3:    ;011-----------------------------;
                                ;29223          M[TEMP6]_R[R4].RR.24,            ;READ was ok, get TEMP6 data
U 10D3, 0086,62B7,0025,0047,0125,6  ;29224      NEXT/DS.DIVP.UNP.2.END
                                ;29225
                                ;29226 10D4:    ;100-----------------------------;
                                ;29227          PC_ZEXT(M[TEMP0])+Q,SIZE[BYTE], ;DR=0, PSL<V> & FPD are ok, take trap
U 10D4, 0480,0019,0000,0487,00FB,9  ;29228      NEXT/IE.FLT.DEC.DBZ
                                ;29229 ;=
                                ;29230
                                ;29231 =0
                                ;29232 DS.DIVP.UNP.2.END:
                                ;29233          ;0-------------------------------;
                                ;29234          M[TEMP6]_ZEXT(MB),SIZE[BYTE],    ;TEMP6 is done, mask out GPR's
U 1256, 0886,659E,4000,0047,053D,F  ;29235      PUSH,NEXT/DS.SR.26
                                ;29236
                                ;29237          ;1-------------------------------;
U 1257, 0080,05BE,4034,C487,011F,E  ;29238      PC_R[R3],NEXT/DS.DIVP.REE.2      ;Fix PC to point to base of DV,
                                ;29239                                           ;reenter inst.
```

```
                        ;29240          ;*************************************************************************
                        ;29241          ; Continuation of Code 3 unpack
                        ;29242          ;*************************************************************************
                        ;29243 =00
                        ;29244 DS.DIVP.UNP.3:
                        ;29245          ;00-----------------------------;
U 11B4, 0586,4C37,0030,0847,011B,5      ;29246          M[TEMP4]_ZLIT0[1]               ;TEMP4 is zero, make it a legal 1
                        ;29247
                        ;29248          ;01-----------------------------;
                        ;29249          M[TEMP5]_MB.AND.ZLIT0[1F],      ;TEMP5 must be 0-1F, unpack DR, [DV],
U 11B5, 0186,5C12,0030,F847,0533,C      ;29250          PUSH,NEXT/DS.DIVP.UNP.C0.034    ;and PC
                        ;29251
                        ;29252          ;10-----------------------------;
                        ;29253          M[TEMP6]_R[R4].RR.24,           ;Ok, get TEMP6 data, mask out GPR's
U 11B6, 0886,62B7,0025,0047,053D,F      ;29254          PUSH,NEXT/DS.SR.26
                        ;29255
                        ;29256          ;11-----------------------------;
                        ;29257          M[TEMP6]_ZEXT(MB),SIZE[BYTE],   ;Fix TEMP6, reenter, anything pending?
U 11B7, 0C86,659E,4B00,18E7,010D,8      ;29258          NEXT/DS.DIVP.REE.3,IP.TS?
                        ;29259
                        ;29260
                        ;29261
                        ;29262          ;*************************************************************************
                        ;29263          ; Continuation of Code 4 unpack
                        ;29264          ;*************************************************************************
                        ;29265 =0
                        ;29266 DS.DIVP.UNP.4:
                        ;29267          ;0-----------------------------;
                        ;29268          M[TEMP5]_MB.AND.ZLIT0[1F],      ;TEMP5 is fixed & legal, unpack DR,
U 1258, 0186,5C12,0030,F847,0533,C      ;29269          PUSH,NEXT/DS.DIVP.UNP.C0.034    ;[DV], and PC
                        ;29270
                        ;29271          ;1-----------------------------;
U 1259, 0486,05BE,4035,0047,0133,B      ;29272          M[TEMP0]_R[R4]                  ;Restore VA
                        ;29273
                        ;29274          ;-----------------------------;
                        ;29275          VA_ZEXT(M[TEMP0])+R[R5],
U 133B, 0080,0015,0005,44A7,0125,A      ;29276          SIZE[BYTE]
                        ;29277
                        ;29278 =0
                        ;29279          ;0-----------------------------;
                        ;29280          M[TEMP6]_R[R4].RR.24,           ;TEMP6 data in byte#0, mask out
U 125A, 0886,62B7,0025,0047,053D,F      ;29281          PUSH,NEXT/DS.SR.26              ;GPR's
                        ;29282
                        ;29283          ;1-----------------------------;
                        ;29284          M[TEMP6]_ZEXT(MB),SIZE[BYTE],   ;TEMP6 is done, reenter inst.
U 125B, 0086,659E,4000,0047,0130,D      ;29285          NEXT/DS.DIVP.REE.4
```

```
;29286 .TOC     "       Decimal String                  :       DIVP common unpack routine"
;29287
;29288          ;*************************************************************************
;29289          ; DS.DIVP.UNP.CO.034 is the DIVP common unpacking routine for int.
;29290          ; codes 0, 3, and 4.
;29291          ;
;29292          ;       Input           TEMP6           Copy of GPR2
;29293          ;                       GPR0            Packed values for each int. code
;29294          ;                       GPR1            DR base address
;29295          ;                       GPR2            Packed values for each int. code
;29296          ;                       GPR3            DV Base address
;29297          ;                       GPR4            Packed values for each int. code
;29298          ;                       (SP)            Points to bottom of saved [DV]
;29299          ;                                       on stack
;29300          ;
;29301          ;       Resources       TEMP0-TEMP3     Used to reread DR, misc. data
;29302          ;                       STEPC           Used as counter by DS.READ
;29303          ;                       VA              For reads
;29304          ;                       QREG            RNUM copy
;29305          ;                       RNUM            MTEMP pointer
;29306          ;
;29307          ;       Output          TEMP1           Packed value, last read DV byte
;29308          ;                                       from memory.
;29309          ;                       RTEMP8-RTEMP11  DR
;29310          ;                       MTEMP8-MERRCOD  [DV] as stored on stack
;29311          ;                                       *MERRCOD is not needed after any
;29312          ;                                       BUS exception*
;29313          ;                       GPR2            DR(LDNLZ) (again)
;29314          ;                       PC              Restored as DV pointer
;29315          ;                       FLAGS           _byte#0 of GPR0 (needed for int.
;29316          ;                                       codes #0 & 4, but done for code
;29317          ;                                       3 anyway)
;29318          ;
;29319          ;       Subroutines     DS.READ.DIVP.DR
;29320          ;
;29321          ;       RETURNs         +1              Everything ok
;29322          ;                       NEVER           If BUS exception or int. pending
;29323          ;                                       occurs.  In such a case, either
;29324          ;                                       in DS.READ or this routine,
;29325          ;                                       R[R2]_M[TEMP6],NEXT/IE.PACK.DONE
;29326          ;                                       Also if DR=0 during unpack, in
;29327          ;                                       which case PC is restored from
;29328          ;                                       PCBACK+deltaPC and divide-by-
;29329          ;                                       zero trap is taken.
;29330          ;
;29331          ;*************************************************************************
```

```
                        ;29332 DS.DIVP.UNP.CO.034:
                        ;29333          ;--------------------------------;
U 133C, 0486,02B7,0005,0047,0102,A      ;29334          M[TEMP0]_R[R4].RR.8             ;Get PC offset into DV
                        ;29335
                        ;29336 102A:
                        ;29337          ;--------------------------------;
                        ;29338          PC_ZEXT(M[TEMP0])+R[R3],        ;PC is reset
U 102A, 0880,0015,0004,C487,0533,A      ;29339          SIZE[BYTE],PUSH         ;RETRUN [+6]
                        ;29340
                        ;29341          ;----- PSUEDO SUBROUTINE ------ ;ALSO CALLED BY LOC 10CA
                        ;29342
                        ;29343 133A:
                        ;29344 DS.DIVP.UNP.2:
                        ;29345          ;--------------------------------;
U 133A, 0086,02B7,0015,0047,0133,E      ;29346          M[TEMP0]_R[R4].RR.16            ;Get VA offset
                        ;29347
                        ;29348          ;--------------------------------;
                        ;29349          STEPC_D_SEXT(M[TEMP0]).SR.1,    ;STEPC_D_L/2 for DR read
U 133E, 0480,0815,A08D,8107,0000,6      ;29350          SIZE[BYTE],RETURN [+6]          ;THIS IS NOT REALLY A SUBROUTINE
                        ;29351                                          ;BUT WE NEEDED 2 EXTRA WORDS
                        ;29352
                        ;29353 ;=000
                        ;29354 1030:    ;000----------------------------;
                        ;29355          VA_D+R[R1]+1,PUSH,              ;Point VA past DR, read it,
U 1030, 0480,00A1,0034,44A7,053C,0      ;29356          NEXT/DS.READ.DIVP.DR            ;RETURN +2, +3, or +4
                        ;29357
                        ;29358 ;=010
                        ;29359 1032:    ;010----------------------------;
                        ;29360          R[R2]_M[TEMP6],                 ;BUS exception or int. pending,
U 1032, 0484,6592,4034,8047,00FE,2      ;29361          NEXT/IE.PACK.DONE               ;Fix GPR's, everything is ok for
                        ;29362                                          ;future unpack
                        ;29363
                        ;29364 1033:    ;011----------------------------;
                        ;29365          VA_R[SP],                       ;Ok, set up to read [DV] from stack
U 1033, 0C80,05BE,4037,84A7,0133,F      ;29366          NEXT/DS.DIVP.UNP.CO.034.C
                        ;29367
                        ;29368 1034:    ;100----------------------------;
                        ;29369          PC_ZEXT(M[TEMP0])+Q,SIZE[BYTE], ;DR=0, PSL<V> & FPD are ok, take trap
U 1034, 0480,0019,0000,0487,00FB,9      ;29370          NEXT/IE.FLT.DEC.DBZ
                        ;29371 ;=
                        ;29372
                        ;29373
                        ;29374 DS.DIVP.UNP.CO.034.C:
                        ;29375          ;--------------------------------;
U 133F, 0880,05BE,4034,0307,0134,0      ;29376          FLAGS_R[R0]                     ;Load FLAGS image, necessary for codes
                        ;29377                                          ;#0 and 4, but #3 gets it anyway
                        ;29378
                        ;29379          ;--------------------------------;
                        ;29380          RNUM_Q_ZLIT0[11.],              ;Point RNUM & QREG to highest MTEMP
U 1340, 0D81,DC37,1030,5847,0104,F      ;29381          NEXT7DS.DIVP.UNP.CO.034LP
```

```
                                ;29382          ;*************************************************************************
                                ;29383          ; =00111 is RNUM=7, we are finished looping.  =01111 is there is
                                ;29384          ; more to read, and =01111 is also where we first enter loop.
                                ;29385          ; =10011 is RETURN+4 from BUS exception during read.
                                ;29386          ;*************************************************************************
                                ;29387 =00011
                                ;29388 =00111
                                ;29389 DS.DIVP.UNP.CO.034LO:
                                ;29390          ;00111-------------------------;
                                ;29391          M[TEMP2]_R[RO].RR.8,            ;[DV] is read, TEMP2_TEMP2 data in
U 1047, 0C86,22B7,0004,0047,0134,4 ;29392       NEXT/DS.DIVP.UNP.CO.034EX       ;low byte
                                ;29393
                                ;29394 =01111
                                ;29395 DS.DIVP.UNP.CO.034LP:
                                ;29396          ;01111-------------------------;
                                ;29397          READ,SIZE[LONG],VA_VA+4,        ;Read (first)(next) lw. of [DV]
U 104F, 0080,0036,4020,0450,0134,1 ;29398       NEXT/DS.DIVP.UNP.CO.034LC
                                ;29399
                                ;29400 =10011
                                ;29401          ;10011-------------------------;
                                ;29402          R[R2]_M[TEMP6],                 ;BUS exception, GPR's are ok for future
U 1053, 0484,6592,4034,8047,00FE,2 ;29403       NEXT/IE.PACK.DONE               ;unpack, bye...
                                ;29404 =
                                ;29405
                                ;29406
                                ;29407 DS.DIVP.UNP.CO.034LC:
                                ;29408          ;-------------------------------;
U 1341, 0C85,2592,4030,0047,0134,2 ;29409       R[TEMPO]_M[MDR]                 ;Get data into proper MTEMP
                                ;29410
                                ;29411          ;-------------------------------;
U 1342, 0087,05BE,4030,0047,0134,3 ;29412       M[TEMP.R]_R[TEMPO]
                                ;29413
                                ;29414          ;-------------------------------;
                                ;29415          RNUM_Q_Q-1,WB<5-0>?,            ;Dec. MTEMP pointer, is it =7?
U 1343, 0081,D0BB,123D,8047,0904,7 400* ;29416  NEXT7DS.DIVP.UNP.CO.034LO        ;( are we done? )
                                ;29417
                                ;29418
                                ;29419 DS.DIVP.UNP.CO.034EX:
                                ;29420          ;-------------------------------;
U 1344, 0486,259E,4000,0047,0134,5 ;29421       M[TEMP2]_ZEXT(MB),SIZE[BYTE]    ;TEMP2 is done
                                ;29422
                                ;29423          ;-------------------------------;
U 1345, 0486,12B7,0021,8047,0134,6 ;29424       M[TEMP1]_R[TEMP6].RR.24         ;TEMP1 data in low byte
                                ;29425
                                ;29426          ;-------------------------------;
                                ;29427          M[TEMP1]_ZEXT(MB),SIZE[BYTE],   ;TEMP1 is done
U 1346, 0886,159E,4080,0047,0000,1 ;29428       RETURN [T]
```

```
                        ;29429 .TOC "  Decimal String                    : CVTLP"
                        ;29430
                        ;29431 ;*************************************************************************
                        ;29432 ;       CVTLP   src.rl, dstlen.rw, dstaddr.ab
                        ;29433 ;
                        ;29434 ;       Input           Q               Source
                        ;29435 ;                       MDR             Destination length
                        ;29436 ;
                        ;29437 ;       Resources       TEMP0           Save source
                        ;29438 ;                       TEMP1           # of significant digits in result.
                        ;29439 ;                       TEMP2           Destinaion length (not zero extended).
                        ;29440 ;                       TEMP3,TEMP4     0 for writing out leading 0's
                        ;29441 ;                       TEMP5,TEMP6     Result
                        ;29442 ;                       RNUM            Used to write result from TEMP3-7
                        ;29443 ;                       PL              Used to skip leading 0 bits in source
                        ;29444 ;                       STEPC           Convert loop control.
                        ;29445 ;                       FLAGS           PSL<NZVC>
                        ;29446 ;
                        ;29447 ;       Output          R0              Zero
                        ;29448 ;                       R1              Zero
                        ;29449 ;                       R2              Zero
                        ;29450 ;                       R3              Destination address
                        ;29451 ;*************************************************************************
                        ;29452
                        ;29453 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;29454 =0
                        ;29455 DS.CVTLP:
                        ;29456         ;0-------------------------------;
                        ;29457         R[TEMP9]_Q Q_D,                 ; SAVE SRC
                        ;29458         LOD INC BRA?,                   ;
U 031E, 0C84,002C,71B2,4047,0412,0      ;29459         PUSH,NEXT/OS.ADD                ; EVALUATE DESTINATION ADDRESS OPERAND
                        ;29460
                        ;29461         ;1-------------------------------;
                        ;29462         M[TEMP0]_R[TEMP9],              ; SAVE SOURCE AND CHECK SIGN.
U 031F, 0486,05BE,46F2,4047,00D3,1      ;29463         WB<31-30>?                      ;
                        ;29464
                        ;29465 .REGION/DECMAL.R3L,DECMAL.R3H
                        ;29466 =01
                        ;29467 DS.CVTLP.10:
                        ;29468         ;01------------------------------; SOURCE IS POSITIVE,
                        ;29469         M[TEMP5]_R[ZERO],               ; INITIALIZE RESULT REGISTER.
U 0D31, 0486,55BE,403D,8047,00D3,2      ;29470         NEXT/DS.CVTLP.20                ;
                        ;29471
                        ;29472         ;11------------------------------; SOURCE IS NEGATIVE,
                        ;29473         M[TEMP0]_-MB,SET FLAG3,         ; CHANGE - SOURCE TO + FOR CONVERT
U 0D33, 005E,0003,003D,8047,00D3,1      ;29474         NEXT/DS.CVTLP.10                ;
                        ;29475
                        ;29476 DS.CVTLP.20:
                        ;29477         ;-------------------------------;
                        ;29478         PL_MSS M[TEMP0],                ; SKIP LEADING 0 BITS IN SOURCE
U 0D32, 0C80,09C2,4A7D,8047,00D5,C      ;29479         WX.NE.0?                        ; CHECK FOR ZERO SOURCE.
```

```
                                ;29480 =0       ;0------------------------------;
                                ;29481          M[TEMP6]_R[ZERO],CLEAR FLAG3,   ;
U 0D5C, 041E,65BE,403D,8047,00D6,3 ;29482       NEXT/DS.CVTLP.45                ;
                                ;29483
                                ;29484          ;1------------------------------; LOAD THE NUMBER OF THE
U 0D5D, 0086,6B37,0030,0047,00D4,C ;29485       M[TEMP6]_PL                     ; MOST SIGNIFICANT BIT INTO STEPC
                                ;29486
                                ;29487          ;------------------------------;
U 0D4C, 0580,6C11,0030,0907,00D4,E ;29488       STEPC_M[TEMP6]+ZLIT0[1]         ;
                                ;29489
                                ;29490          ;------------------------------;
U 0D4E, 0D86,6C37,0030,0047,00D5,2 ;29491       M[TEMP6]_ZLIT0[0]               ; INITIALIZE RESULT REGISTER
                                ;29492
                                ;29493          ;------------------------------;
                                ;29494          M[TEMP0]_MB.RR.P,               ; SKIP LEADING 0 BITS IN SOURCE.
U 0D52, 0C86,0177,02B0,0047,00D6,0 ;29495       WB<0>?,NEXT/DS.CVTLP.30         ; BRANCH ON MOST SIGNIFICANT BIT SET
                                ;29496 =0
                                ;29497 DS.CVTLP.LOOP:
                                ;29498          ;0------------------------------;
                                ;29499          M[TEMP0]_MB.RL.1,               ; SHIFT NEXT BIT INTO BIT ZERO.
U 0D5E, 0486,0301,C2BD,8047,08D6,0 355* ;29500  WB<0>?,NEXT/DS.CVTLP.30         ; AND BRANCH ON IT.
                                ;29501
                                ;29502          ;1------------------------------;
                                ;29503          INTPEND OR TIMER?,              ;
U 0D5F, 0480,0036,4AF0,0047,04F7,0 ;29504       PUSH,NEXT/IE.SERV.IP.TS2        ;
                                ;29505 =0
                                ;29506 DS.CVTLP.30:
                                ;29507          ;0------------------------------; BIT IS ZERO, ADD ZERO TO RESULT,
U 0D60, 0480,0036,4330,0047,00D6,2 ;29508       DBZ STEPC?,NEXT/DS.CVTLP.40     ; DECREMENT COUNT OF BITS REMAINING
                                ;29509
                                ;29510          ;1------------------------------; BIT IS ONE, ADD ONE TO THE RESULT,
U 0D61, 0186,6C91,4020,8047,00D5,6 ;29511       M[TEMP6]_(MB+ZLIT24[10]).BCD    ; SKIPPING THE SIGN POSITION
                                ;29512
                                ;29513          ;------------------------------;
                                ;29514          M[TEMP5]_(MB+R[ZERO]+ALKC).BCD, ; ADD CARRY FOR DOUBLE RESULT.
U 0D56, 0086,5041,432D,8047,00D6,2 ;29515       DBZ STEPC?                      ; DECREMENT COUNT OF BITS REMAINING
                                ;29516 =0
                                ;29517 DS.CVTLP.40:
                                ;29518          ;0------------------------------;
                                ;29519          M[TEMP6]_(MB+R[TEMP6]).BCD,     ; DOUBLE LOWER HALF OF RESULT
U 0D62, 0486,6001,4021,8047,00D5,8 ;29520       NEXT/DS.CVTLP.50                ;
                                ;29521
                                ;29522 DS.CVTLP.45:
                                ;29523          ;1------------------------------;
                                ;29524          M[TEMP2]_Q,                     ; MOVE DESTINATION LENGTH TO A TEMP
U 0D63, 0486,203A,44FD,8047,00D6,4 ;29525       FLAG3?,NEXT/DS.CVTLP.SIGN       ; BRANCH ON SIGN OF SOURCE.
                                ;29526
                                ;29527 DS.CVTLP.50:
                                ;29528          ;------------------------------;
                                ;29529          M[TEMP5]_(MB+R[TEMP5]+ALKC).BCD, ; DOUBLE UPPER HALF OF RESULT
U 0D58, 0486,5041,4B21,58E7,00D5,E ;29530       IP.TS?,NEXT/DS.CVTLP.LOOP       ;
```

```
                                        ;29531 =0
                                        ;29532 DS.CVTLP.SIGN:
                                        ;29533          ;0------------------------------;
                                        ;29534          M[TEMP6]_MB.OR.ZLIT24[12.],     ; STORE PLUS IN RESULT.
U 0D64, 0D86,6C92,4030,6047,00D7,1      ;29535          NEXT/DS.CVTLP.SRCLEN            ;
                                        ;29536
                                        ;29537          ;1------------------------------;
U 0D65, 0986,6C92,4030,6847,00D7,1      ;29538          M[TEMP6]_MB.OR.ZLIT24[13.]      ; STORE MINUS IN RESULT.
                                        ;29539
                                        ;29540 DS.CVTLP.SRCLEN:
                                        ;29541          ;------------------------------;
U 0D71, 0480,5592,4A70,0047,00D3,4      ;29542          WB_M[TEMP5],WX.NE.0?            ; SEE IF RESULT HAS MORE THAN 7 DIGITS.
                                        ;29543
                                        ;29544 =00      ;00------------------------------; LONGWORD RESULT
                                        ;29545          R[TEMP1]_M[TEMP6].BCDSWP,       ; COMPUTE # OF NON ZERO DIGITS.
U 0D34, 0C84,6637,0030,4047,04D8,F      ;29546          PUSH,NEXT/DS.CVTLP.NON.0.DIGIT  ;
                                        ;29547
                                        ;29548          ;01------------------------------; DOUBLE WORD RESULT
                                        ;29549          R[TEMP1]_M[TEMP5].BCDSWP,       ; COMPUTE # OF NON ZERO DIGITS.
U 0D35, 0C84,5637,0030,4047,04D8,F      ;29550          PUSH,NEXT/DS.CVTLP.NON.0.DIGIT  ;
                                        ;29551
                                        ;29552 DS.CVTLP.CHECK.OVFLW:
                                        ;29553          ;10------------------------------;
                                        ;29554          WB_ZEXT(M[TEMP2])-R[TEMP1],     ;Is destination length < result length?
                                        ;29555          WB<31-30>?,SIZE[WORD],
U 0D36, 0480,2014,06D0,4047,08D3,9 349* ;29556          NEXT/DS.CVTLP.70
                                        ;29557
                                        ;29558          ;11------------------------------;
                                        ;29559          M[TEMP1]_MB+ZLIT0[8],           ; CORRECT FOR DOUBLE WORD RESULT
U 0D37, 0186,1C11,0030,4047,00D3,6      ;29560          NEXT/DS.CVTLP.CHECK.OVFLW       ;
                                        ;29561
                                        ;29562 =01
                                        ;29563 DS.CVTLP.70:
                                        ;29564          ;11------------------------------;
                                        ;29565          WB_SEXT(M[TEMP2]).ANDNOT.ZLIT0[1F], ;Result <= Destination
                                        ;29566          SIZE[WORD],
U 0D39, 0180,2C1F,8A10,F847,00D6,8      ;29567          WX.EQ.0?,NEXT/DS.CVTLP.80
                                        ;29568
                                        ;29569          ;11------------------------------;
                                        ;29570          WB_ZEXT(M[TEMP2]),SIZE[WORD],   ;Result > Destination, check for even or
U 0D3B, 0048,259E,4290,0047,00D6,6      ;29571          SET FLAG1,WB<0>?                ;odd dest. length (for high nibble)
```

```
                                                    B 9
                                ;29572 =0
                                ;29573          ;0------------------------------;
                                ;29574          WB_ZEXT(M[TEMP2]),SIZE[WORD],   ;Even dest., which nibble do we zero?
U 0D66, 0480,259E,4210,0047,00D7,0  ;29575      WB<5-0>?,NEXT/DS.CVTLP.OVFLW
                                ;29576
                                ;29577          ;1------------------------------;
                                ;29578          VA_R[TEMP0]_M[MDR],             ; CHECK THAT FIRST BYTE OF DESTINATION
U 0D67, 0C85,2592,4030,04A7,00D6,A  ;29579      NEXT/DS.CVTLP.83                ; CAN BE WRITTEN.
                                ;29580
                                ;29581 =110000
                                ;29582 DS.CVTLP.OVFLW:
                                ;29583          ;110000--------------------------;
                                ;29584          M[TEMP6]_MB.ANDNOT.ZLIT24[0F0], ;Dest. length=0, clear high nibble of
U 0D70, 0586,6C93,8037,8047,00D6,9  ;29585      NEXT/DS.CVTLP.81                ;of destination
                                ;29586
                                ;29587 =110010
                                ;29588          ;110010--------------------------;
                                ;29589          M[TEMP6]_MB.ANDNOT.ZLIT16[0F0], ;Dest. length=2, ditto
U 0D72, 0586,6D13,8037,8047,00D6,9  ;29590      NEXT/DS.CVTLP.81
                                ;29591
                                ;29592 =110100
                                ;29593          ;110100--------------------------;
                                ;29594          M[TEMP6]_MB.ANDNOT.ZLIT8[0F0],  ;Dest. length=4, ditto
U 0D74, 0186,6D93,8037,8047,00D6,9  ;29595      NEXT/DS.CVTLP.81
                                ;29596
                                ;29597 =110110
                                ;29598          ;110110--------------------------;
                                ;29599          M[TEMP6]_MB.ANDNOT.ZLIT0[0F0],  ;Dest. length=6, ditto
U 0D76, 0186,6C13,8037,8047,00D6,9  ;29600      NEXT/DS.CVTLP.81
                                ;29601
                                ;29602 =111000
                                ;29603          ;111000--------------------------;
                                ;29604          M[TEMP5]_MB.ANDNOT.ZLIT24[0F0], ;Dest. length=8, ditto
U 0D78, 0586,5C93,8037,8047,00D6,9  ;29605      NEXT/DS.CVTLP.81
                                ;29606 =
                                ;29607
                                ;29608 =0
                                ;29609 DS.CVTLP.80:
                                ;29610          ;0------------------------------;
U 0D68, 0C80,0036,4030,0047,00FF,8  ;29611      NEXT/IE.OPER.FAULT              ;
                                ;29612
                                ;29613 DS.CVTLP.81:
                                ;29614          ;1------------------------------;
U 0D69, 0C85,2592,4030,04A7,00D6,A  ;29615      VA_R[TEMP0]_M[MDR]              ; CHECK THAT FIRST BYTE OF DESTINATION
                                ;29616 =0
                                ;29617 DS.CVTLP.83:
                                ;29618          ;0------------------------------;
                                ;29619          PROBE WRITE?,SIZE[BYTE],        ;
U 0D6A, 0480,0036,4780,005D,0D68,4 479*  ;29620 PUSH,NEXT/MM.PRB.WRITE.SIZ        ; CAN BE WRITTEN.
                                ;29621
                                ;29622          ;1------------------------------;
                                ;29623          M[TEMP3]_R[ZERO],               ; CLEAR TO WRITE LEADING 0'S
U 0D6B, 0886,358E,483D,98E7,00D6,C  ;29624      IP.TS?                          ;
```

```
                                ;29625 =0
                                ;29626          ;0------------------------------;
                                ;29627          M[TEMP4]_R[ZERO],               ; CLEAR TO WRITE LEADING 0'S
U 0D6C, 0086,45BE,403D,8047,00D7,3 ;29628       NEXT/DS.CVTLP.WRT.DST           ;
                                ;29629
                                ;29630          ;1------------------------------;
                                ;29631          INTPEND OR TIMER?,              ;
U 0D6D, 0480,0036,4AF0,0047,04F7,0 ;29632       PUSH,NEXT/IE.SERV.IP.TS2        ;
                                ;29633
                                ;29634 DS.CVTLP.WRT.DST:
                                ;29635          ;-------------------------------;
U 0D73, 0585,DC37,0030,3047,00D7,5 ;29636       RTEMP7_RNUM_ZLIT0[6]            ;
                                ;29637
                                ;29638          ;-------------------------------;
                                ;29639          M[TEMP2]_ZEXT(MB).SR.1,
U 0D75, 0886,2015,801D,8047,00D7,7 ;29640       SIZE[WORD]
                                ;29641
                                ;29642          ;-------------------------------;
                                ;29643          VA_M[TEMP0]+R[TEMP2]+1,         ;
                                ;29644          SET FLAG2,                      ; WILL BE CLEARED ON FIRST NON ZERO
U 0D77, 0450,0081,0030,84A7,00D7,8 ;29645       NEXT/DS.CVTLP.90                ; DESTINATION DIGIT ACTUALLY WRITTEN
                                ;29646 =0*
                                ;29647 DS.CVTLP.WRT.LONG:
                                ;29648          ;0*------------------------------;
                                ;29649          RNUM_R[TEMP7]_RB-1,             ;
                                ;29650          SET MM.NOINT,                   ;
U 0D38, 0465,DE7D,0031,C047,00D7,8 ;29651       NEXT/DS.CVTLP.90                ;
                                ;29652
                                ;29653          ;1*------------------------------;
                                ;29654          RNUM_R[TEMP7]_RB-1,             ;
                                ;29655          SET MM.NOINT,                   ;
U 0D3A, 0C65,DE7D,0471,C047,0088,0 ;29656       MM.NOINT?                       ;
                                ;29657
                                ;29658 880:     ;0********FORCE ADDRESS*********; LAST WORD OF DESTINATION
                                ;29659          WB_M[TEMP.R+1].ANDNOT.ZLIT24[0F], ; CHECK FOR ZERO IGNORING THE SIGN
U 0880, 0181,1C93,8A30,7847,00D7,A ;29660       WX.EQ.0?,NEXT/DS.CVTLP.85       ;
                                ;29661
                                ;29662 881:     ;1********FORCE ADDRESS*********; NOT LAST WORD OF DESTINATION
                                ;29663          WB_M[TEMP.R+1],                 ; CHECK FOR ZERO ALL 8 DIGITS
U 0881, 0881,1592,4A30,0047,08D7,A 322* ;29664  WX.EQ.0?,NEXT/DS.CVTLP.85       ;
                                ;29665
                                ;29666 =0
                                ;29667 FREE.0D6E:
                                ;29668          ;0******LOCATION NOT USED ******; LAST WORD OF DESTINATION
                                ;29669          WB_M[TEMP.R].ANDNOT.ZLIT24[0F], ; CHECK FOR ZERO IGNORING THE SIGN
U 0D6E, 0581,0C93,8A30,7847,00D7,A ;29670       WX.EQ.0?,NEXT/DS.CVTLP.85       ;
                                ;29671
                                ;29672 FREE.0D6F:
                                ;29673          ;1*****LOCATION NOT USED********; NOT LAST WORD OF DESTINATION
U 0D6F, 0C81,0592,4A30,0047,08D7,A 322* ;29674  WB_M[TEMP.R],WX.EQ.0?            ; CHECK FOR ZERO ALL 8 DIGITS=0
                                ;29675
                                ;29676 -0
                                ;29677 DS.CVTLP.85:
                                ;29678          ;0------------------------------;
                                ;29679          VA_M[VA]-ZLIT0[4],CLEAR FLAG2,  ; DESTINATION IS NON ZERO CLEAR PSL<Z>
```

```
U 0D7A, 0511,BC10,0030,24A7,00D7,9      ;29680          NEXT/DS.CVTLP.93                ;
                                        ;29681
                                        ;29682 DS.CVTLP.90:
                                        ;29683          ;1-----------------------------;
U 0D7B, 0581,BC10,0030,24A7,00D7,9      ;29684          VA_M[VA]-ZLIT0[4]               ;
                                        ;29685
                                        ;29686 DS.CVTLP.93:
                                        ;29687          ;-----------------------------;
                                        ;29688          WB_M[VA]-R[TEMP0],              ;
U 0D79, 0C81,B000,0B60,0047,08D3,D 371* ;29689          SIGND CMP?,SIZE[LONG]           ;
                                        ;29690
                                        ;29691 =01      ;01----------------------------;
                                        ;29692          WRITE R[TEMP.R],SIZE[LONG],     ;
                                        ;29693          FLAG1?,                         ; IF OVERFLOW CHECK FOR ZERO DESTINATION
U 0D3D, 0080,05BE,45EC,05D8,00D3,8      ;29694          NEXT/DS.CVTLP.WRT.LONG          ;
                                        ;29695
                                        ;29696          ;11----------------------------;
                                        ;29697          VA_M[VA]+ZLIT0[3],              ;
U 0D3F, 0981,BC11,0030,1CA7,00D8,A      ;29698          NEXT/DS.CVTLP.100               ;
                                        ;29699 =0*
                                        ;29700 DS.CVTLP.WRT.BYTE:
                                        ;29701          ;0*----------------------------;
                                        ;29702          WRITE R[TEMP.R],SIZE[BYTE],     ;
U 0D3C, 0C80,05BE,400C,05D8,00D7,E      ;29703          NEXT/DS.CVTLP.95                ;
                                        ;29704
                                        ;29705          ;1*----------------------------;
                                        ;29706          WRITE R[TEMP.R],SIZE[BYTE],     ;
U 0D3E, 0480,05BE,444C,05D8,00D7,C      ;29707          MM.NOINT?                       ;
                                        ;29708
                                        ;29709 =0       ;0----------------------------;
                                        ;29710          WB_M[TEMP.R].AND.ZLIT0[0F0],    ;
                                        ;29711          SET MM.NOINT,                   ;
U 0D7C, 0961,0C12,0A77,8047,00D7,E      ;29712          WX.NE.0?,NEXT/DS.CVTLP.95       ;
                                        ;29713
                                        ;29714          ;1-----------------------------;
                                        ;29715          WB_M[TEMP.R].AND.ZLIT0[0FF],    ;
                                        ;29716          SET MM.NOINT,                   ;
U 0D7D, 0561,0C12,0A77,F847,00D7,E      ;29717          WX.NE.0?                        ;
                                        ;29718 =0
                                        ;29719 DS.CVTLP.95:
                                        ;29720          ;0-----------------------------;
                                        ;29721          VA_M[VA]-ZLIT0[1],              ;
                                        ;29722          SET MM.NOINT,                   ;
U 0D7E, 0161,BC10,0030,0CA7,00D8,A      ;29723          NEXT/DS.CVTLP.100               ;
                                        ;29724
                                        ;29725          ;1-----------------------------;
                                        ;29726          VA_M[VA]-ZLIT0[1],              ;
U 0D7F, 0911,BC10,0030,0CA7,00D8,A      ;29727          CLEAR FLAG2                     ;
```

```
                                        ;29728 DS.CVTLP.100:
                                        ;29729          ;--------------------------------;
                                        ;29730          WB_M[VA]-R[TEMP0],               ;
U 0D8A, 0481,B000,0B60,0047,08D4,1 371* ;29731          SIGND CMP?,SIZE[LONG]            ;
                                        ;29732
                                        ;29733 =01      ;01--------------------------------;
                                        ;29734          R[TEMP.R]_RB.RL.8,               ;
                                        ;29735          FLAG1?,                          ; IF OVERFLOW CHECK FOR ZERO
U 0D41, 0C84,02B7,05EC,0047,00D3,C      ;29736          NEXT/DS.CVTLP.WRT.BYTE           ; DESTINATION
                                        ;29737
                                        ;29738          ;11--------------------------------;
U 0D43, 0884,05B7,0034,0047,00D8,B      ;29739          R[R0]_0                          ;
                                        ;29740
                                        ;29741          ;--------------------------------;
U 0D8B, 0484,05B7,0034,4047,00D8,C      ;29742          R[R1]_0                          ;
                                        ;29743
                                        ;29744          ;--------------------------------;
U 0D8C, 0C84,05B7,0034,8047,00D8,D      ;29745          R[R2]_0
                                        ;29746
                                        ;29747          ;--------------------------------;
                                        ;29748          WB_R[TEMP9],CCOP1,SIZE[LONG],    ;
U 0D8D, 0880,05BE,4562,4847,00D4,4      ;29749          FLAG1 (FLAG2.XOR.FLAG3)?
                                        ;29750
                                        ;29751 =00
                                        ;29752 DS.CVTLP.DONE:
                                        ;29753          ;00--------------------------------; NO OVERFLOW
U 0D44, 0484,0592,4134,C047,003F,9      ;29754          R[R3]_M[TEMP0],IRD1              ;Condition codes are set, copy dest.
                                        ;29755                                           ;address & **IRD1**
                                        ;29756
                                        ;29757          ;01--------------------------------; NO OVERFLOW
U 0D45, 0484,0592,4134,C047,003F,9      ;29758          R[R3]_M[TEMP0],IRD1              ;Condition codes are set, copy dest.
                                        ;29759                                           ;address & **IRD1**
                                        ;29760
                                        ;29761          ;10--------------------------------; OVERFLOW AND PSL<N,Z> = 00 OR 11
U 0D46, 0418,0036,4030,0047,00D4,7      ;29762          CLEAR FLAG3                      ; CHANGE TO 00 OR 01
                                        ;29763
                                        ;29764          ;11--------------------------------; OVERFLOW AND PSL<N,Z> = 01 OR 10
U 0D47, 0886,2036,4030,0387,00D8,E      ;29765          M[TEMP2]_FLAGS                   ; USE FLAGS TO SET CONDTION CODES
                                        ;29766
                                        ;29767          ;--------------------------------;
U 0D8E, 0480,2592,4030,00A7,00D4,4      ;29768          CC_M[TEMP2],NEXT/DS.CVTLP.DONE   ;CC is set, IRD1
                                        ;29769
                                        ;29770
                                        ;29771 DS.CVTLP.NON.0.DIGIT:
                                        ;29772          ;--------------------------------;
U 0D8F, 0C80,19C2,403D,8047,00D9,0      ;29773          PL_MSS M[TEMP1]                  ;
                                        ;29774
                                        ;29775          ;--------------------------------;
U 0D90, 0086,1B3D,803D,8047,00D9,1      ;29776          M[TEMP1]_(R[ZERO]+PL).SR.1       ;
                                        ;29777
                                        ;29778          ;--------------------------------;
                                        ;29779          M[TEMP1]_MB.SR.1,                ;
U 0D91, 0886,1001,80BD,8047,0000,2      ;29780          RETURN [+2]                      ;
```

```
                                ;29781 .TOC "  Decimal String                    : CVTPL"
                                ;29782
                                ;29783 ;*****************************************************************************
                                ;29784 ;       CVTPL   srclen.rw, srcaddr.ab, dst.wl
                                ;29785 ;
                                ;29786 ;       Input           MDR             Source length
                                ;29787 ;
                                ;29788 ;       Resources       TEMP0           Next source digit
                                ;29789 ;                       TEMP2           Intermediate results
                                ;29790 ;                       TEMP5           Save source address
                                ;29791 ;                       TEMP6           OF used to extract nibbles
                                ;29792 ;                       TEMP7           Save ZEXT(source length)
                                ;29793 ;                       MTEMP10         Save VA
                                ;29794 ;                       MDR             Inputing source digits
                                ;29795 ;                       STEPC
                                ;29796 ;                       FLAG1           Set if overflow
                                ;29797 ;                       FPDOFFSET       Set to 0
                                ;29798 ;
                                ;29799 ;       Output          R0              Zero
                                ;29800 ;                       R1              Source address
                                ;29801 ;                       R2              Zero
                                ;29802 ;                       R3              Zero
                                ;29803 ;*****************************************************************************
                                ;29804
                                ;29805 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;29806 DS.CVTPL:
                                ;29807         ;-------------------------------;
                                ;29808         R[TEMP7]_ZEXT(M[MDR]),          ; SAVE ZEXT(SOURCE LENGTH)
                                ;29809         LOD INC BRA?,SIZE[WORD],        ;
U 03BA, 0485,259E,4191,C047,0012,0 ;29810      NEXT/OS.ADD                     ; EVALUATE SOURCE ADDRESS OPERAND
                                ;29811
                                ;29812 DS.CVTPL.02:
                                ;29813         ;-------------------------------;
U 03BB, 0086,25BE,403D,8047,00D9,2 ;29814      M[TEMP2]_R[ZERO]                ; INITIALIZE RESULT
                                ;29815
                                ;29816 .REGION/DECMAL.R3L,DECMAL.R3H
                                ;29817         ;-------------------------------;
U 0D92, 0986,6C37,0030,7847,00D9,3 ;29818      M[TEMP6]_ZLIT0[0F]              ; USED TO EXTRACT NIBBLES
                                ;29819
                                ;29820         ;-------------------------------;
                                ;29821         WB_M[TEMP7].ANDNOT.ZLIT0[1F],   ; CHECK SOURCE LENGTH
U 0D93, 0580,7C13,8A30,F847,00D8,0 ;29822      WX.EQ.0?                        ;
                                ;29823
                                ;29824 =0      ;0-------------------------------;
U 0D80, 0C80,0036,4030,0047,00FF,8 ;29825      NEXT/IE.OPER.FAULT              ;
                                ;29826
                                ;29827         ;1-------------------------------;
U 0D81, 0D80,7C11,8030,1107,00D9,4 ;29828      STEPC_(M[TEMP7]+ZLIT0[2]).SR.1  ; LOAD # OF BYTES IN SOURCE
                                ;29829
                                ;29830         ;-------------------------------;
U 0D94, 0C85,2592,4031,44A7,00D9,5 ;29831      VA_R[TEMP5]_M[MDR]              ; LOAD AND SAVE SOURCE ADDRESS
                                ;29832
                                ;29833         ;-------------------------------;
U 0D95, 0C86, 5BE,4031,4047,00D9,6 ;29834      M[TEMP10]_R[TEMP5]              ; SAVE VA
```

```
                                ;29835          ;--------------------------------;
U 0D96, 0980,0EF6,4030,2047,00D8,2  ;29836      PL_[4]                           ; USED FOR ROTATING
                                ;29837
                                ;29838 =0
                                ;29839 DS.CVTPL.SKIP.ZEROS:
                                ;29840          ;0-------------------------------;
                                ;29841          READ,SIZE[BYTE],                 ; READ SOURCE BYTE
                                ;29842          VA_M[TEMP10]_MB+ZLIT0[1],        ;
U 0D82, 0586,AC11,0300,0CB0,00D8,4  ;29843      DBZ STEPC?,NEXT/DS.CVTPL.20      ;
                                ;29844
                                ;29845          ;1-------------------------------;
U 0D83, 0886,7036,4030,0187,00D9,7  ;29846      M[TEMP7]_STEPC                   ; SAVE # OF BYTES REMAINING IN SOURCE
                                ;29847
                                ;29848          ;--------------------------------;
                                ;29849          M[TEMP7]_MB.AND.ZLIT0[1F],       ; MASK OF GARBAGE FROM SOURCING STEPC
U 0D97, 0586,7C12,0030,F847,00D9,8  ;29850      NEXT/DS.CVTPL.CHECK.OVFLW        ;
                                ;29851 =0
                                ;29852 DS.CVTPL.20:
                                ;29853          ;0-------------------------------;
                                ;29854          R[TEMP0]_ZEXT(M[MDR]),           ; CHECK FOR ZERO
                                ;29855          SIZE[BYTE],WX.NE.0?,             ;
U 0D84, 0885,259E,4A40,0047,00D8,2  ;29856      NEXT/DS.CVTPL.SKIP.ZEROS         ; SKIP LEADING ZERO'S
                                ;29857
                                ;29858          ;1-------------------------------;
                                ;29859          D_(M[MDR].RR.P).AND.R[TEMP6],    ;
U 0D85, 0881,217E,2031,8047,00D8,8  ;29860      NEXT/DS.CVTPL.END.CONVERT        ;
```

```
                                ;29861          ;*****************************************************************************
                                ;29862          ; We've read a nonzero BYTE from the source.  The first overflow
                                ;29863          ; check is made from the number of significant bytes/digits.
                                ;29864          ;*****************************************************************************
                                ;29865
                                ;29866 DS.CVTPL.CHECK.OVFLW:
                                ;29867          ;-------------------------------;
                                ;29868          WB_M[TEMP7]-ZLIT0[5],           ; CHECK IF LESS THAN 10 SIGNIFICANT
U 0D98, 0180,7C10,0B60,2847,08D4,8 413*  ;29869          SIGND CMP DEF?,SIZE[LONG]       ; DIGITS IN SOURCE.
                                ;29870
                                ;29871 =00      ;00-----------------------------;
                                ;29872          MTEMP0_(MDR.RR.P).AND.R[TEMP6], ;
U 0D48, 044F,217E,0031,8047,00D9,9       ;29873          SET FLAG1,NEXT/DS.CVTPL.CONVERT ;
                                ;29874
                                ;29875          ;01-----------------------------;
                                ;29876          WB_M[TEMP0]-ZLIT0[3],           ;Check if greater than 2,200,000,000
U 0D49, 0D80,0C10,06F0,1847,08D4,D 377*  ;29877          WB<31-30>?,NEXT/DS.CVTPL.30     ;( 2!20!00!00!00!0S )
                                ;29878
                                ;29879 DS.CVTPL.LOOP:
                                ;29880          ;10-----------------------------;
                                ;29881          MTEMP0_(MDR.RR.P).AND.R[TEMP6], ;
U 0D4A, 0C87,217E,0031,8047,00D9,9       ;29882          NEXT/DS.CVTPL.CONVERT           ;
                                ;29883
                                ;29884          ;11-----------------------------;
                                ;29885          D_(M[MDR].RR.P).AND.R[TEMP6],   ; GET LEAST SIGNIFICANT DIGIT
U 0D4B, 0881,217E,2031,8047,00D8,8       ;29886          NEXT/DS.CVTPL.END.CONVERT       ;
                                ;29887
                                ;29888 =01
                                ;29889 DS.CVTPL.30:
                                ;29890          ;01-----------------------------;
                                ;29891          MTEMP0_(MDR.RR.P).AND.R[TEMP6],
U 0D4D, 044F,217E,0031,8047,00D9,9       ;29892          SET FLAG1,NEXT/DS.CVTPL.CONVERT
                                ;29893
                                ;29894          ;11-----------------------------;
U 0D4F, 0C87,217E,0031,8047,00D9,9       ;29895          MTEMP0_(MDR.RR.P).AND.R[TEMP6]  ;
                                ;29896
                                ;29897 DS.CVTPL.CONVERT:
                                ;29898          ;-------------------------------;
U 0D99, 0886,2001,C030,0047,00D9,A       ;29899          M[TEMP2]_(MB+R[TEMP0]).SL.1     ; ADD NEXT DIGIT AND SAVE RESULT*2
                                ;29900
                                ;29901          ;-------------------------------;
                                ;29902          M[TEMP2]_MB+(R[TEMP2].ASL.2),   ; FINISH MULTIPLY BY 10 (2*(2X)+2X)
U 0D9A, 0086,25D1,0420,8047,00D8,6       ;29903          FLAG0?                          ; RIGHT OR LEFT NIBBLE?
                                ;29904
                                ;29905 =0       ;0-----------------------------;
                                ;29906          MTEMP0_MDR.AND.R[TEMP6],        ; GET RIGHT NIBBLE
U 0D86, 0C47,2002,0031,8047,00D9,9       ;29907          SET FLAG0,NEXT/DS.CVTPL.CONVERT ; SET TO GET LEFT NEXT TIME
                                ;29908
                                ;29909          ;1-----------------------------;
                                ;29910          READ,SIZE[BYTE],CLEAR FLAG0,    ;Read next BYTE & get left nibble,
                                ;29911          VA_M[TEMP10]_MB+ZLIT0[1],       ;FLAG0 0 to get right nibble next time,
U 0D87, 0506,AC11,0300,0CB0,00D4,A       ;29912          DBZ STEPC?,NEXT/DS.CVTPL.LOOP   ;branch on # of source BYTEs remaining
```

```
                                ;29913          ;*************************************************************************
                                ;29914          ; End of CVTPL loop.  Make final check for overflow & WRITE dest.
                                ;29915          ;*************************************************************************
                                ;29916 =0
                                ;29917 DS.CVTPL.END.CONVERT:
                                ;29918          ;0-----------------------------;
                                ;29919          ALUS_BCD SIGN.ZERO(M[MDR]),     ; LATCH SIGN OF SOURCE
                                ;29920          LOD INC BRA?,                   ;
U 0D88, 0081,2002,41BD,80C7,0414,0      ;29921          PUSH,NEXT/OS.WRT1               ; EVALUATE DESTINATION OPERAND.
                                ;29922
                                ;29923          ;1-----------------------------;
U 0D89, 0CEE,C5BE,403D,8047,00D9,B      ;29924          M[FPDOFFSET]_R[ZERO],SET FPD    ;
                                ;29925
                                ;29926          ;-----------------------------;
U 0D9B, 0484,05B7,0B74,9847,00D5,1      ;29927          R[R2]_0,BCD SIGN.ZERO?          ;
                                ;29928
                                ;29929 =01      ;01----------------------------;
                                ;29930          R[TEMP2]_D+RB,CCOP1,SIZE[LONG], ; RESULT IS POSITIVE
U 0D51, 0084,0021,06E0,8847,08D5,9 338* ;29931          WB<31-30>?,NEXT/DS.CVTPL.50     ; DO FINAL CHECK FOR OVERFLOW
                                ;29932
                                ;29933          ;11----------------------------; RESULT IS NEGATIVE
U 0D53, 0884,0021,0030,8047,00D9,C      ;29934          R[TEMP2]_D+RB                   ; ADD LEAST SIGNIFICANT DIGIT TO RESULT
                                ;29935
                                ;29936          ;-----------------------------;
                                ;29937          M[TEMP2]_-MB,CCOP1,SIZE[LONG],  ; RESULT IS NEGATIVE OF MAGNITUDE.
U 0D9C, 0486,2003,06ED,8847,08D5,5 344* ;29938          WB<31-30>?                      ; DO FINAL CHECK FOR OVERFLOW
                                ;29939
                                ;29940          ;*************************************************************************
                                ;29941          ; Negative source overflow checks.  =11 is present data is ok, check
                                ;29942          ; for overflow detected during initial source read.  =01 is present
                                ;29943          ; data overflowed (and TEMP2 is infinite precision low significant
                                ;29944          ; digits) OR data is zero (src. =-0), check for overflow during initial
                                ;29945          ; src. read.
                                ;29946          ;*************************************************************************
                                ;29947 =01
                                ;29948          ;01----------------------------;
                                ;29949          WB_M[TEMP2],WX.NE.0?,           ;Overflow if data isn't zero
U 0D55, 0880,2592,4A70,0047,00D5,A      ;29950          NEXT/DS.CVTPL.NEG.OVR
                                ;29951
                                ;29952          ;11----------------------------;
                                ;29953          R[R3]_0,FLAG1?,                 ; CHECK IF OVERFLOW DETECTED EARLIER
U 0D57, 0084,05B7,05F4,C047,00D4,0      ;29954          NEXT/DS.CVTPL.NO.OVFLW          ;
```

```
                        ;29955          ;*************************************************************************
                        ;29956          ; Overlapping splits.  Enter at:
                        ;29957          ; 1) DS.CVTPL.50: on a WB<31-30>? BUT from positive source overflow
                        ;29958          ;    check.  =01 is data is ok, check for previous overflow.
                        ;29959          ;            =11 is data overflowed.
                        ;29960          ; 2) DS.CVTPL.NEG.OVR: on a WX.NE.0? BUT from negative source
                        ;29961          ;    overflow check.  =10 is data is ok (-0), check for previous ov.
                        ;29962          ;                     =11 is data overflowed.
                        ;29963          ;*************************************************************************
                        ;29964 =00
                        ;29965 =01
                        ;29966 DS.CVTPL.50:
                        ;29967          ;01-----------------------------;
                        ;29968          R[R3]_0,FLAG1?,                 ; CHECK IF OVERFLOW DETECTED EARLIER
U 0D59, 0084,05B7,05F4,C047,00D4,0      ;29969          NEXT/DS.CVTPL.NO.OVFLW          ;
                        ;29970
                        ;29971 DS.CVTPL.NEG.OVR:
                        ;29972          ;10-----------------------------;
                        ;29973          R[R3]_0,FLAG1?,                 ; CHECK IF OVERFLOW DETECTED EARLIER
U 0D5A, 0084,05B7,05F4,C047,00D4,0      ;29974          NEXT/DS.CVTPL.NO.OVFLW          ;
                        ;29975
                        ;29976          ;11-----------------------------;
U 0D5B, 0884,05B7,0034,C047,00D4,2      ;29977          R[R3]_0,NEXT/DS.CVTPL.OVERFLOW  ;Overflow
                        ;29978 =0*
                        ;29979 DS.CVTPL.NO.OVFLW:
                        ;29980          ;0*-----------------------------;
                        ;29981          R[R1]_M[TEMP5],                 ; NO OVERFLOW
U 0D40, 0484,5592,4034,4047,0082,0      ;29982          NEXT/DS.CVTPL.CLEAR.R0          ;
                        ;29983
                        ;29984 DS.CVTPL.OVERFLOW:
                        ;29985          ;1*-----------------------------;
U 0D42, 0084,5592,4034,48E7,0082,0      ;29986          R[R1]_M[TEMP5],SET V            ; OVERFLOW OCCURED
                        ;29987
                        ;29988 0820:
                        ;29989 DS.CVTPL.CLEAR.R0:
                        ;29990          ;*********FORCE ADDRESS*********;
                        ;29991          R[R0]_0,SET MM.NOINT,           ;
U 0820, 0064,05B7,0034,0047,00D5,0      ;29992          NEXT/DS.CVTPL.WRT.DST           ;
                        ;29993
                        ;29994 0FF0:    ;*********FORCE ADDRESS*********;
                        ;29995          ;-----------------------------;
U 0FF0, 0884,05B7,0034,0047,00D5,0      ;29996          R[R0]_0                         ;
                        ;29997 =0**
                        ;29998 DS.CVTPL.WRT.DST:
                        ;29999          ;0**----------------------------;
                        ;30000          WRITE NOTREG,SIZE[LONG],        ;
                        ;30001          R[DST.R]_M[TEMP2],CLEAR FPD,    ;
U 0D50, 00E4,2592,402C,45DA,0000,2      ;30002          NEXT/GL.NOP.IRD1                ;
                        ;30003
                        ;30004          ;1**----------------------------; THE WRITE DID NOT COMPLETE
U 0D54, 0885,A592,4034,8047,0134,7      ;30005          R[R2]_M[PC]                     ; SAVE PC
                        ;30006
                        ;30007 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                        ;30008          ;-----------------------------;
U 1347, 0885,B592,4034,0047,0134,8      ;30009          R[R0]_M[VA]                     ; SAVE DESTINATION ADDRESS
```

```
                                ;30010          ;--------------------------------;
                                ;30011          R[R3]_M[TEMP2],                 ; SAVE RESULT
U 1348, 0484,2592,4034,C047,00FE,2  ;30012      NEXT/IE.PACK.DONE               ;
                                ;30013
                                ;30014
                                ;30015 .TOC "  Decimal String                 :       FPD unpack"
                                ;30016
                                ;30017 DS.CVTPL.IRD1:
                                ;30018          ;--------------------------------;
U 1349, 0484,05B7,0134,0047,003F,9  ;30019      R[R0]_0,IRD1                    ;
                                ;30020
                                ;30021 .REGION/IRD1.R1L,IRD1.R1H
                                ;30022 =000
                                ;30023 DS.CVTPL.UNPACK:
                                ;30024          ;000-----------------------------;
U 0358, 0C80,05BE,4034,8487,0134,A  ;30025      PC_R[R2]                        ; RESTORE SAVED PC
                                ;30026 =
                                ;30027 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;30028          ;--------------------------------;
U 134A, 0886,2036,4030,0087,0134,B  ;30029      M[TEMP2]_PSL                    ;Write back CC to reassert arithmetic
                                ;30030                                          ;trap if we set V
                                ;30031
                                ;30032          ;--------------------------------;
U 134B, 0C80,2592,4030,00A7,0134,C  ;30033      CC_M[TEMP2]
                                ;30034
                                ;30035          ;--------------------------------;
U 134C, 0886,C5B7,0030,0047,0134,D  ;30036      M[FPDOFFSET]_0                  ; MIGHT FAULT AGAIN, BE PREPARED
                                ;30037
                                ;30038          ;--------------------------------;
U 134D, 0880 05BE,4034,04A7,0134,E  ;30039      VA_R[R0]                        ; RESTORE DESTINATION ADDRESS
                                ;30040
                                ;30041          ;--------------------------------;
U 134E, 0464,05B7,0034,8047,0107,2  ;30042      R[R2]_0,SET MM.NOINT            ;
                                ;30043
                                ;30044 =0**     ;0**-----------------------------;
                                ;30045          WRITE R[R3],SIZE[LONG],         ;
U 1072, 04E0,05BE,4024,C5D8,0134,F  ;30046      CLEAR FPD,NEXT/DS.CVTPL.CLR.R3  ;
                                ;30047
                                ;30048          ;1**-----------------------------; THE WRITE DID NOT COMPLETE
                                ;30049          R[R2]_M[PC],                    ; SAVE PC
U 1076, 0085,A592,4034,8047,00FE,2  ;30050      NEXT/IE.PACK.DONE               ;
                                ;30051
                                ;30052 DS.CVTPL.CLR.R3:
                                ;30053          ;--------------------------------;
U 134F, 0084,05B7,0034,C047,0134,9  ;30054      R[R3]_0,NEXT/DS.CVTPL.IRD1       ;
```

```
                                ;30055 .TOC "  Decimal String                    : CVTPT"
                                ;30056
                                ;30057 ;*************************************************************************
                                ;30058 ;       CVTPT   srclen.rw, srcaddr.ab, tbladdr.ab, dstlen.rw, dstaddr.ab
                                ;30059 ;
                                ;30060 ;       Input           MDR             Source length
                                ;30061 ;
                                ;30062 ;       Resources       TEMP1           Zext(source length)
                                ;30063 ;                       TEMP2           Source address
                                ;30064 ;                       TEMP3           Table address
                                ;30065 ;                       TEMP4           Zero extended destination length
                                ;30066 ;                       TEMP7           Address of last destination byte
                                ;30067 ;                       FPDOFFSET       Set to 0A
                                ;30068 ;
                                ;30069 ;       Output          R0              Zero
                                ;30070 ;                       R1              Source address
                                ;30071 ;                       R2              Zero
                                ;30072 ;                       R3              Destination address
                                ;30073 ;
                                ;30074 ;       Subroutines     DS.CVTPN
                                ;30075 ;                       DS.TABLE.ADDR
                                ;30076 ;                       DS.CVTXX.SETCC
                                ;30077 ;*************************************************************************
                                ;30078
                                ;30079 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;30080
                                ;30081 DS.CVTPT:
                                ;30082         ;-------------------------------;
                                ;30083         R[TEMP1] ZEXT(M[MDR]),SIZE[WORD], ; SAVE  ZEXT(SOURCE LENGTH)
                                ;30084         LOD INC BRA?,                     ;
U 03BC, 0485,259E,4190,4047,0012,0 ;30085      NEXT/OS.ADD                       ; EVALUATE SOURCE ADDRESS OPERAND
                                ;30086 =000
                                ;30087 DS.CVTPT.02:
                                ;30088         ;000----------------------------;
                                ;30089         R[TEMP2] M[MDR],                  ; SAVE SOURCE ADDRESS
                                ;30090         LOD INC BRA?,                     ;
U 0360, 0885,2592,41B0,8047,0412,0 ;30091      PUSH,NEXT/OS.ADD                  ; EVALUATE TABLE ADDRESS
                                ;30092
                                ;30093         ;001----------------------------;
                                ;30094         R[TEMP3] M[MDR],                  ; SAVE TABLE ADDRESS
                                ;30095         LOD INC BRA?,                     ;
U 0361, 0485,2592,41B0,C047,0410,0 ;30096      PUSH,NEXT/OS.RED                  ; EVALUATE DESTINATION LENGTH OPERAND
                                ;30097
                                ;30098         ;010----------------------------;
                                ;30099         R[TEMP4] ZEXT(M[MDR]),SIZE[WORD], ; SAVE ZEXT(DESTINATION LENGTH)
                                ;30100         LOD INC BRA?,                     ;
U 0362, 0C85,259E,4191,0047,0412,0 ;30101      PUSH,NEXT/OS.ADD                  ; EVALUATE DESTINATION ADDRESS OPERAND
```

```
                                ;30102 DS.CVTPT.RESTART:
                                ;30103          ;011----------------------------;
                                ;30104          M[FPDOFFSET]_ZLIT0[0A],SET FPD, ;
U 0363, 05EE,CC37,0030,5047,0528,0 ;30105       PUSH,NEXT/DS.CVTPN              ; CONVERT PACKED TO NUMERIC
                                ;30106
                                ;30107          ;100----------------------------;
                                ;30108          R[R2]_0,                        ;
U 0364, 0484,05B7,0034,8047,0537,E ;30109       PUSH,NEXT/DS.TABLE.ADDR         ; LOAD TABLE ADDRESS OF LAST BYTE
                                ;30110
                                ;30111 DS.CVTPT.SETCC:
                                ;30112          ;101----------------------------;
                                ;30113          R[R2]_0,                        ;
U 0365, 0484,05B7,05F4,8047,010A,8 ;30114       FLAG1?,NEXT/DS.CVTPT.20         ; IF OVERFLOW, CLEAR PSL<Z>
                                ;30115
                                ;30116          ;110----------------------------; IF OVERFLOW, CLEAR PSL<Z>
                                ;30117          READ,SIZE[BYTE],                ; GET BYTE FROM TABLE
U 0366, 0C80,7002,45CD,84B0,010A,0 ;30118       VA_M[TEMP7],FLAG1?              ; LOAD ADDRESS OF LAST DESTINATION BYTE
                                ;30119 =
                                ;30120
                                ;30121 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;30122 =0*
                                ;30123 DS.CVTPT.10:
                                ;30124          ;0*-----------------------------; NO OVERFLOW
                                ;30125          WRITE M[MDR],SIZE[BYTE],        ; WRITE LAST BYTE
                                ;30126          BCD SIGN.ZERO?,                 ;
U 10A0, 0081,2592,4B40,1DD8,011E,8 ;30127       NEXT/DS.CVTXX.SETCC             ;
                                ;30128
                                ;30129          ;1*-----------------------------; OVERFLOW, CLEAR PSL<Z>
                                ;30130          WRITE M[MDR],SIZE[BYTE],        ; WRITE LAST BYTE
                                ;30131          BCD SIGN.ZERO?,                 ;
U 10A2, 0011,2592,4B40,1DD8,011E,8 ;30132       CLEAR FLAG2,NEXT/DS.CVTXX.SETCC ;
                                ;30133
                                ;30134 =0*
                                ;30135 DS.CVTPT.20:
                                ;30136          ;0*-----------------------------; NO OVERFLOW
                                ;30137          BCD SIGN.ZERO?,                 ;
U 10A8, 0480,0036,4B70,1847,011E,8 ;30138       NEXT/DS.CVTXX.SETCC             ;
                                ;30139
                                ;30140          ;1*-----------------------------; OVERFLOW, CLEAR PSL<Z>
                                ;30141          BCD SIGN.ZERO?,                 ;
U 10AA, 0410,0036,4B70,1847,011E,8 ;30142       CLEAR FLAG2,NEXT/DS.CVTXX.SETCC ;
```

```
                                ;30143 .TOC "  Decimal String                    : CVTTP"
                                ;30144
                                ;30145 ;*****************************************************************************
                                ;30146 ;       CVTTP   srclen.rw, srcaddr.ab, tbladdr.ab, dstlen.rw, dstaddr.ab
                                ;30147 ;
                                ;30148 ;       Input           MDR             Source length
                                ;30149 ;
                                ;30150 ;       Resources       TEMP0           Converted sign from table
                                ;30151 ;                       TEMP1           Zext(source length)
                                ;30152 ;                       TEMP2           Source address
                                ;30153 ;                       TEMP3           Table address
                                ;30154 ;                       TEMP4           Zero extended destination length
                                ;30155 ;                       TEMP6           Source byte
                                ;30156 ;                       TEMP7           Address of last destination byte
                                ;30157 ;                       MDR             Save byte from table
                                ;30158 ;                       FLAG0           Set to indicate 0 length destination
                                ;30159 ;
                                ;30160 ;       Output          R0              Zero
                                ;30161 ;                       R1              Source address
                                ;30162 ;                       R2              Zero
                                ;30163 ;                       R3              Destination address
                                ;30164 ;
                                ;30165 ;       Subroutines     DS.CVTNP
                                ;30166 ;                       DS.TABLE.ADDR
                                ;30167 ;                       DS.CVTXX.SETCC.07
                                ;30168 ;                       DS.CVTXX.SETCC.06
                                ;30169 ;*****************************************************************************
                                ;30170
                                ;30171 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;30172 DS.CVTTP:
                                ;30173          ;-------------------------------;
                                ;30174          R[TEMP1] ZEXT(M[MDR]),SIZE[WORD], ; SAVE  ZEXT(SOURCE LENGTH)
                                ;30175          LOD INC BRA?,                     ;
U 03BD, 0485,259E,4190,4047,0012,0 ;30176       NEXT/OS.ADD                       ; EVALUATE SOURCE ADDRESS OPERAND
                                ;30177 =000
                                ;30178 DS.CVTTP.02:
                                ;30179          ;000----------------------------;
                                ;30180          R[TEMP2] M[MDR],                  ; SAVE SOURCE ADDRESS
                                ;30181          LOD INC BRA?,                     ;
U 0368, 0885,2592,4180,8047,0412,0 ;30182       PUSH,NEXT/OS.ADD                  ; EVALUATE TABLE ADDRESS
                                ;30183
                                ;30184          ;001----------------------------;
                                ;30185          R[TEMP3] M[MDR],                  ; SAVE TABLE ADDRESS
                                ;30186          LOD INC BRA?,                     ;
U 0369, 0485,2592,4180,C047,0410,0 ;30187       PUSH,NEXT/OS.RED                  ; EVALUATE DESTINATION LENGTH OPERAND
```

```
                                ;30188          ;010----------------------------;
                                ;30189          R[TEMP4]_ZEXT(M[MDR]),SIZE[WORD], ; SAVE ZEXT(DESTINATION LENGTH)
U 036A, 0885,259E,4A51,0047,0037,0 ;30190       WX.NE.0?                        ; CHECK IF ZERO LENGTH DESTINATION
                                ;30191 =
                                ;30192 370:     ;000----------------------------;
U 0370, 0446,4081,003D,8047,0037,1 ;30193       M[TEMP4]_MB+1,SET FLAG0         ; PRETEND DESTINATION LENGTH IS ONE
                                ;30194
                                ;30195 371:     ;001----------------------------;
                                ;30196          LOD INC BRA?,                   ;
U 0371, 0C80,0036,41B0,0047,0412,0 ;30197       PUSH,NEXT/OS.ADD                ; EVALUATE DESTINATION ADDRESS OPERAND
                                ;30198 372:
                                ;30199 DS.CVTTP.RESTART:
                                ;30200          ;010----------------------------;
                                ;30201          M[FPDOFFSET]_ZLIT0[0A].SET FPD, ;
U 0372, 0DEE,CC37,0030,5047,0535,E ;30202       PUSH,NEXT/DS.CVTNP              ; CONVERT NUMERIC TO PACKED
                                ;30203
                                ;30204 373:     ;011----------------------------;
                                ;30205          R[TEMP6]_ZEXT(XB) PC_PC+1,      ; GET LAST SOURCE BYTE
U 0373, 0C85,759E,4001,A047,0537,E ;30206       PUSH,NEXT/DS.TABLE.ADDR         ; LOAD TABLE ADDRESS OF LAST BYTE
                                ;30207 374:
                                ;30208 DS.CVTTP.ZERO.SRC:
                                ;30209          ;100----------------------------; ZERO LENGTH SOURCE
                                ;30210          M[TEMP6]_R[ZERO],CLEAR FLAG0,   ; CLEAR ZERO LENGTH DESTINATION FLAG
U 0374, 0406,65BE,403D,8047,010E,1 ;30211       NEXT/DS.CVTTP.PLUS              ; AND WRITE +0 DESTINATION
                                ;30212
                                ;30213 375:     ;101----------------------------;
                                ;30214          READ,SIZE[BYTE],                ; GET BYTE FROM TABLE
U 0375, 0C80,7002,400D,84B0,0135,0 ;30215       VA_M[TEMP7]                     ; LOAD ADDRESS OF LAST DESTINATION BYTE
                                ;30216
                                ;30217 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;30218 DS.CVTTP.SETCC:
                                ;30219          ;---------------------------------;
                                ;30220          R[TEMP6]_ZEXT(M[MDR]),          ; EXTRACT SIGN AND LAST DIGIT
                                ;30221          SIZE[BYTE],                     ;
U 1350, 0C85,259E,4001,80C7,0135,1 ;30222       ALUS_BCD SIGN.ZERO              ; SAVE SIGN AND ZERO FOR BRANCHING
                                ;30223
                                ;30224          ;---------------------------------;
                                ;30225          WB_ZLIT0[9F]-M[TEMP6],          ; CHECK IF LAST DIGIT IS LEGAL BCD
U 1351, 0180,6C13,06F4,F847,091B,9 377* ;30226  WB<31-30>?                       ;
                                ;30227
                                ;30228 =01      ;01---------------------------------;
                                ;30229          BCD CHECK M[TEMP6]?,            ; CHECK IF LEGAL SIGN NIBBLE.
U 1189, 0880,67FD,49BD,8047,0925,C 387* ;30230  NEXT/DS.CVTTP.LEGAL.SIGN         ;
                                ;30231
                                ;30232          ;11-----------------------------;
U 11BB, 0C80,0036,4030,0047,00FF,8 ;30233       NEXT/IE.OPER.FAULT              ; LAST DIGIT IS NOT LEGAL BCD DIGIT
                                ;30234 =0
                                ;30235 DS.CVTTP.LEGAL.SIGN:
                                ;30236          ;0------------------------------;
U 125C, 0C80,0036,4030,0047,00FF,8 ;30237       NEXT/IE.OPER.FAULT              ; NOT LEGAL BCD SIGN NIBBLE
                                ;30238
                                ;30239          ;1------------------------------;
U 125D, 0880,0036,4430,0047,0125,F ;30240       FLAG0?                          ; CHECK IF DESTINATION LENGTH IS ZERO
```

```
                                        ;30241 =0       ;0--------------------------------;  CC_0 AS HARMLESS SIDE EFFECT OF BRANCH
                                        ;30242          CC_M[TEMP6]_MB.AND.ZLIT0[0F0],  ;  SAVE LAST DIGIT FOR LATER USE.
                                        ;30243          BCD SIGN.ZERO(DEF)?,SIZE[LONG], ;  BRANCH ON SIGN AND LAST DIGIT=0?
U 125E,      5C12,0B67,80A7,010E,0      ;30244          NEXT/DS.CVTTP.CHK.SIGN          ;
                                        ;30245
                                        ;30246          ;1--------------------------------;
                                        ;30247          WB_M[MDR].AND.ZLIT0[0F0],       ;  CHECK IF LAST SOURCE DIGIT IS ZERO
                                        ;30248          CLEAR FLAG0,                    ;  CLEAR FLAG FOR SETTING CONDITION CODES
U 125F, 0101,2C12,0A37,8047,0126,0      ;30249          WX.EQ.0?                        ;
                                        ;30250
                                        ;30251 =0       ;0--------------------------------;  CLEAR DIGIT FOR WRITING 1 BYTE OF
                                        ;30252          MDR_M[TEMP6].ANDNOT.ZLIT0[0F0], ;  ZERO LENGTH PACKED DESTINATION
U 1260, 0948,6C13,8037,8467,00BF,0      ;30253          SET FLAG1,NEXT/DS.CVTTP.DLEN.0  ;
                                        ;30254
                                        ;30255          ;1--------------------------------;  CC_0 AS HARMLESS SIDE EFFECT OF BRANCH
                                        ;30256          CC_M[TEMP6]_MB.AND.ZLIT0[0F0],  ;  SAVE LAST DIGIT FOR LATER USE.
U 1261, 0586,6C12,0B67,80A7,010E,0      ;30257          BCD SIGN.ZERO(DEF)?,SIZE[LONG]  ;  BRANCH ON SIGN AND LAST DIGIT=0?
```

```
                                ;30258 =000
                                ;30259 DS.CVTTP.CHK.SIGN:
                                ;30260          ;000----------------------------; NON ZERO POSITIVE RESULT
U 10E0, 0C10,0036,4030,0047,010E,1  ;30261      CLEAR FLAG2                      ; CLEAR PSL<Z>
                                ;30262
                                ;30263 DS.CVTTP.PLUS:
                                ;30264          ;001----------------------------; POSITIVE RESULT (LAST DIGIT IS 0)
                                ;30265          WRITE M[TEMP6].OR.ZLIT0[0C],     ; WRITE LAST DIGIT AND PREFERRED '+'.
                                ;30266          SIZE[BYTE],CLEAR FLAG3,          ; CLEAR PSL<N>
U 10E1, 0118,6C12,4000,65D8,011E,9  ;30267      NEXT/DS.CVTXX.SETCC.06           ;
                                ;30268
                                ;30269 DS.CVTTP.MINUS:
                                ;30270          ;010----------------------------; NON ZERO NEGATIVE RESULT
                                ;30271          WRITE M[TEMP6].OR.ZLIT0[0D],     ; WRITE LAST DIGIT AND PREFERRED '-'.
                                ;30272          SIZE[BYTE],CLEAR FLAG2,          ; CLEAR PSL<Z>
U 10E2, 0510,6C12,4000,6DD8,0109,8  ;30273      NEXT/DS.CVTXX.SETCC.08
                                ;30274
                                ;30275          ;011----------------------------; NEGATIVE RESULT (LAST DIGIT IS 0)
                                ;30276          MDR_ZLIT0[0D],SET FLAG3,         ; SET PSL<N>
U 10E3, 0D58,0C37,04B0,6C67,010E,2  ;30277      FLAG2?,NEXT/DS.CVTTP.MINUS       ; CHECK FOR -0
                                ;30278
                                ;30279 =110     ;110----------------------------; -0 RESULT
                                ;30280          MDR_ZLIT0[0D],CLEAR FLAG3,       ; IF OVERFLOW WRITE -0.
U 10E6, 0D18,0C37,05F0,6C67,011B,D  ;30281      FLAG1?                           ; IF NOT OVERFLOW WRITE +0.
                                ;30282 =
                                ;30283 =01      ;01-----------------------------;
                                ;30284          WRITE M[TEMP6].OR.ZLIT0[0C],     ; WRITE +0 RESULT
                                ;30285          SIZE[BYTE],CLEAR FLAG3,          ; CLEAR PSL<N>
U 11BD, 0118,6C12,4000,65D8,011E,9  ;30286      NEXT/DS.CVTXX.SETCC.06           ;
                                ;30287
                                ;30288          ;11-----------------------------;
                                ;30289          WRITE M[MDR],SIZE[BYTE],         ; WRITE -0 RESULT
U 11BF, 0881,2592,4000,05D8,011E,9  ;30290      NEXT/DS.CVTXX.SETCC.06           ;
                                ;30291
                                ;30292 0BF0:    ;*********FORCE ADDRESS*********;
                                ;30293 DS.CVTTP.DLEN.0:
                                ;30294          ;-------------------------------;
                                ;30295          M[FPDOFFSET]_ZLIT0[2E],          ; SET SPECIAL PACK ROUTINE FOR ZERO
U 0BF0, 0986,CC37,0031,7047,0135,0  ;30296      NEXT/DS.CVTTP.SETCC              ; LENGTH DESTINATION
```

```
                                ;30297 .TOC "  Decimal String                    : CVTPS"
                                ;30298
                                ;30299 ;*************************************************************************
                                ;30300 ;       CVTPS   srclen.rw, srcaddr.ab, dstlen.rw, dstaddr.ab
                                ;30301 ;
                                ;30302 ;       Input           MDR             Source length
                                ;30303 ;
                                ;30304 ;       Resources       TEMP1           Zext(source length)
                                ;30305 ;                       TEMP2           Source address
                                ;30306 ;                       TEMP3           Destination address
                                ;30307 ;                       TEMP4           Zero extended destination length
                                ;30308 ;                       TEMP6           Source bytes
                                ;30309 ;                       TEMP7           Temporary
                                ;30310 ;                       FPDOFFSET       Set to 0A
                                ;30311 ;
                                ;30312 ;       Output          R0              Zero
                                ;30313 ;                       R1              Source address
                                ;30314 ;                       R2              Zero
                                ;30315 ;                       R3              Destination address
                                ;30316 ;
                                ;30317 ;       Subroutines     DS.CVTPN
                                ;30318 ;                       DS.CVTXX.SETCC
                                ;30319 ;*************************************************************************
                                ;30320
                                ;30321 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;30322 DS.CVTPS:
                                ;30323         ;--------------------------------;
                                ;30324         R[TEMP1]_ZEXT(M[MDR]),SIZE[WORD], ; SAVE  ZEXT(SOURCE LENGTH)
                                ;30325         LOD INC BRA?,                     ;
U 03BE, 0485,259E,4190,4047,0012,0 ;30326      NEXT/OS.ADD                       ; EVALUATE SOURCE ADDRESS OPERAND
                                ;30327
                                ;30328 =00
                                ;30329 DS.CVTPS.02:
                                ;30330         ;00--------------------------------;
                                ;30331         R[TEMP2]_M[MDR],                  ; SAVE SOURCE ADDRESS
                                ;30332         LOD INC BRA?,                     ;
U 02A4, 0085,2592,41B0,8047,0410,0 ;30333      PUSH,NEXT/OS.RED                  ; EVALUATE DESTINATION LENGTH OPERAND
                                ;30334
                                ;30335         ;01--------------------------------;
                                ;30336         R[TEMP4]_ZEXT(M[MDR]),SIZE[WORD], ; SAVE ZEXT(DESTINATION LENGTH)
                                ;30337         LOD INC BRA?,                     ;
U 02A5, 0C85,259E,4191,0047,0412,0 ;30338      PUSH,NEXT/OS.ADD                  ; EVALUATE DESTINATION ADDRESS OPERAND
                                ;30339
                                ;30340 DS.CVTPS.RESTART:
                                ;30341         ;10--------------------------------;
U 02A6, 0485,2592,4030,C047,0135,2 ;30342      R[TEMP3]_M[MDR]                   ; SAVE DESTINATION ADDRESS
                                ;30343 =
                                ;30344 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;30345         ;--------------------------------;
U 1352, 0C80,3711,0000,0467,010E,8 ;30346      MDR_M[TEMP3]+CONX(1)              ; SAVE ADDRESS OF FIRST DESTINATION DIGI
```

```
                        ;30347 =000
                        ;30348 =011
                        ;30349 DS.CVTPS.CONVERT:
                        ;30350          ;011----------------------------;
                        ;30351          M[FPDOFFSET]_ZLIT0[0A],SET FPD, ;
U 10EB, 05EE,CC37,0030,5047,0528,0      ;30352          PUSH,NEXT/DS.CVTPN              ; CONVERT PACKED TO NUMERIC
                        ;30353
                        ;30354          ;100----------------------------;
                        ;30355          WRITE CVTPN(M[TEMP6]),          ; WRITE LAST DIGIT
U 10EC, 0880,66F7,0DC0,05D8,010E,D      ;30356          SIZE[BYTE],S<3-0>.NE.0?         ; CHECK LAST DIGIT FOR ZERO
                        ;30357
                        ;30358          ;101----------------------------;
                        ;30359          VA_R[R3]-CONX(1),               ; RESTORE DESTINATION ADDRESS
U 10ED, 0880,073C,0004,C4A7,0135,3      ;30360          NEXT/DS.CVTPS.WRT.SIGN          ;
                        ;30361
                        ;30362 =111     ;111----------------------------;
                        ;30363          VA_R[R3]-CONX(1),               ;
                        ;30364          CLEAR FLAG2,                    ;
U 10EF, 0810,073C,0004,C4A7,0135,3      ;30365          NEXT/DS.CVTPS.WRT.SIGN          ;
                        ;30366
                        ;30367 DS.CVTPS.WRT.SIGN:
                        ;30368          ;----------------------------;
                        ;30369          WB_M[TEMP6],                    ; CHECK SIGN OF SOURCE
U 1353, 0880,6592,4B70,1847,011C,1      ;30370          BCD SIGN.ZERO?                  ;
                        ;30371
                        ;30372 =01      ;01----------------------------;
                        ;30373          WRITE ZLIT0[2B],SIZE[BYTE],     ; STORE LEADING '+'
U 11C1, 0580,0C37,05C1,5DD8,011C,4      ;30374          FLAG1?,NEXT/DS.CVTPS.SETCC      ; IS OVERFLOW FLAG SET?
                        ;30375
                        ;30376          ;11----------------------------;
                        ;30377          WRITE ZLIT0[2D],SIZE[BYTE],     ; STORE LEADING '-'
                        ;30378          SET FLAG3,                      ;
                        ;30379          FLAG1 (FLAG2.XOR.FLAG3)?,       ;
U 11C3, 0D58,0C37,0541,6DD8,011C,4      ;30380          NEXT/DS.CVTPS.SETCC             ; CHECK FOR OVERFLOW AND -0
                        ;30381 =00
                        ;30382 DS.CVTPS.SETCC:
                        ;30383          ;00----------------------------; NO OVERFLOW AND DESTINATION NE 0
U 11C4, 0484,05B7,0034,8047,0135,4      ;30384          R[R2]_0,NEXT/DS.CVTPS.FIX.R3    ;
                        ;30385
                        ;30386          ;01----------------------------; NO OVERFLOW AND DESTINATION = 0
                        ;30387          WRITE ZLIT0[2B],SIZE[BYTE],     ; STORE LEADING '+'
U 11C5, 0580,0C37,0001,5DD8,011C,4      ;30388          NEXT/DS.CVTPS.SETCC             ;
                        ;30389
                        ;30390          ;10----------------------------; OVERFLOW AND DESTINATION NE 0
                        ;30391          R[R2]_0,CLEAR FLAG2,            ; CLEAR PSL<Z>
U 11C6, 0414,05B7,0034,8047,0135,4      ;30392          NEXT/DS.CVTPS.FIX.R3            ;
                        ;30393
                        ;30394          ;11----------------------------; OVERFLOW AND DESTINATION = 0
                        ;30395          R[R2]_0,CLEAR FLAG2,            ; CLEAR PSL<Z>
U 11C7, 0414,05B7,0034,8047,0135,4      ;30396          NEXT/DS.CVTPS.FIX.R3            ;
                        ;30397
                        ;30398 DS.CVTPS.FIX.R3:
                        ;30399          ;----------------------------;
                        ;30400          R[R3]_RB-1,BCD SIGN.ZERO?,      ;
U 1354, 0484,0E7D,0B74,D847,011E,8      ;30401          NEXT/DS.CVTXX.SETCC             ;
```

```
                                ;30402 .TOC "  Decimal String                    : CVTSP"
                                ;30403
                                ;30404 ;*****************************************************************************
                                ;30405 ;       CVTSP   srclen.rw, srcaddr.ab, dstlen.rw, dstaddr.ab
                                ;30406 ;
                                ;30407 ;       Input           MDR             Source length
                                ;30408 ;
                                ;30409 ;       Resources       TEMP1           Zext(source length)
                                ;30410 ;                       TEMP2           Source address
                                ;30411 ;                       TEMP3           Save sign
                                ;30412 ;                       TEMP4           Zero extended destination length
                                ;30413 ;                       TEMP6           Source byte
                                ;30414 ;                       TEMP7           Save destination address
                                ;30415 ;                       MTEMP8          Constant F0F0
                                ;30416 ;                       FPDOFFSET       Set to 0A
                                ;30417 ;
                                ;30418 ;       Output          R0              Zero
                                ;30419 ;                       R1              Source address
                                ;30420 ;                       R2              Zero
                                ;30421 ;                       R3              Destination address
                                ;30422 ;*****************************************************************************
                                ;30423
                                ;30424 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;30425 DS.CVTSP:
                                ;30426         ;-------------------------------;
                                ;30427         R[TEMP1]_ZEXT(M[MDR]),SIZE[WORD], ; SAVE  ZEXT(SOURCE LENGTH)
                                ;30428         LOD INC BRA?,                   ;
U 03BF, 0485,259E,4190,4047,0012,0 ;30429      NEXT/OS.ADD                     ; EVALUATE SOURCE ADDRESS OPERAND
                                ;30430 2A8:
                                ;30431 DS.CVTSP.02:
                                ;30432         ;00********FORCE ADDRESS*********;
                                ;30433         R[TEMP2]_M[MDR],                ; SAVE SOURCE ADDRESS
                                ;30434         LOD INC BRA?,                   ;
U 02A8, 0085,2592,41B0,8047,0410,0 ;30435      PUSH,NEXT/OS.RED                ; EVALUATE DESTINATION LENGTH OPERAND
                                ;30436
                                ;30437 2A9:    ;01-------------------------------;
                                ;30438         R[TEMP4]_ZEXT(M[MDR]),SIZE[WORD], ; SAVE ZEXT(DESTINATION LENGTH)
                                ;30439         LOD INC BRA?,                   ;
U 02A9, 0C85,259E,4191,0047,0412,0 ;30440      PUSH,NEXT/OS.ADD                ; EVALUATE DESTINATION ADDRESS OPERAND
                                ;30441
                                ;30442 2AA:    ;10-------------------------------;
U 02AA, 0885,2592,4031,C047,0135,5 ;30443      R[TEMP7]_M[MDR]                 ; SAVE DESTINATION ADDRESS
                                ;30444
                                ;30445 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;30446         ;-------------------------------;
U 1355, 0C80,2002,403D,84A7,0135,6 ;30447      VA_M[TEMP2]                     ; LOAD SOURCE ADDRESS
                                ;30448
                                ;30449         ;-------------------------------;
                                ;30450         READ,SIZE[BYTE],                ; READ SOURCE SIGN BYTE
U 1356, 0986,2C11,0000,0850,0135,7 ;30451      M[TEMP2]_MB+ZLIT0[1]            ; SAVE FIRST SOURCE DIGIT ADDRESS
                                ;30452
                                ;30453         ;-------------------------------;
U 1357, 0881,27C2,4DFD,8047,011C,8 ;30454      ASCII SIGN(M[MDR])?             ; BRANCH AND SAVE SIGN
```

```
                                ;30455 =00      ;00------------------------------;
                                ;30456          M[TEMP3]_ZLIT0[0D],              ; SAVE BCD -
U 11C8, 0586,3C37,0030,6847,0135,8 ;30457       NEXT/DS.CVTSP.20                 ;
                                ;30458
                                ;30459          ;01------------------------------;
                                ;30460          M[TEMP3]_ZLIT0[0C],              ; SAVE BCD +
U 11C9, 0186,3C37,0030,6047,0135,8 ;30461       NEXT/DS.CVTSP.20                 ;
                                ;30462
                                ;30463 =11      ;11------------------------------;
U 11CB, 0CE0,0036,4030,0047,00FF,8 ;30464       CLEAR FPD,NEXT/IE.OPER.FAULT     ;
                                ;30465
                                ;30466 DS.CVTSP.20:
                                ;30467          ;--------------------------------;
U 1358, 0480,7002,403D,8467,011C,C ;30468       MDR_M[TEMP7]                     ; RESTORE DESTINATION ADDRESS
                                ;30469
                                ;30470          ;*****************************************************************************
                                ;30471          ; CVTNP returns 2 ways:
                                ;30472          ; =01 is normal return.
                                ;30473          ; =10 is either Dest. length=0, Source length=0, or dest. and source
                                ;30474          ; lengths=0.  If <V>image is set, (last) dest. BYTE_sign of source.
                                ;30475          ; If <V>image is clear, (last) dest. BYTE_[0C].
                                ;30476          ;*****************************************************************************
                                ;30477 =00
                                ;30478 DS.CVTSP.RESTART:
                                ;30479          ;00------------------------------;
                                ;30480          M[FPDOFFSET]_ZLIT0[0A],SET FPD,  ;Load FPDOFFSET, convert Sep. to Pack.,
U 11CC, 0DEE,CC37,0030,5047,0535,E ;30481       PUSH,NEXT/DS.CVTNP               ;RETURN +1 or +2
                                ;30482
                                ;30483          ;01------------------------------;
                                ;30484          R[TEMP6]_ZEXT(XB) PC_PC+1,       ;Normal return, get last byte
U 11CD, 0485,759E,4001,A047,0135,A ;30485       NEXT/DS.CVTSP.40
                                ;30486
                                ;30487          ;10------------------------------;Dest. length = 0, Point GPR1 to
U 11CE, 0884,0E7D,0034,4047,0135,9 ;30488       R[R1]_RB-1                       ;source base add.
                                ;30489 =
                                ;30490          ;--------------------------------;
                                ;30491          M[FPDOFFSET]_ZLIT0[18],          ; LOAD OFFSET OF SPECIAL PACK ROUTINE
U 1359, 0586,CC37,05F0,C047,010B,8 ;30492       FLAG1?,NEXT/DS.CVTSP.0.DEST      ; DID WE HAVE OV?
```

```
                                ;30493 DS.CVTSP.40:
                                ;30494          ;--------------------------------;
U 135A, 0480,8002,2031,8047,0135,8  ;30495      D_M[TEMP8].AND.R[TEMP6]          ;
                                ;30496
                                ;30497          ;--------------------------------;
                                ;30498          WB_D.XOR.ZLIT0[30],              ; CHECK FOR LEGAL ASCII DIGIT
U 135B, 0580,0C33,4A31,8047,0126,2  ;30499      WX.EQ.0?                         ;
                                ;30500
                                ;30501 =0       ;0------------------------------;
U 1262, 0C80,0036,4030,0047,00FF,8  ;30502      NEXT/IE.OPER.FAULT               ;Inst. will pack up to service this
                                ;30503
                                ;30504          ;1------------------------------;
                                ;30505          M[TEMP6]_CVTNP(MB),              ; CONVERT THE LAST DIGIT
U 1263, 0086,67FD,49BD,8047,0926,4 387*  ;30506 BCD CHECK?                        ; MAKE SURE RESULT DIGIT IS 0-9
                                ;30507
                                ;30508 =0       ;0------------------------------;
                                ;30509          MDR_M[TEMP6].OR.R[TEMP3],        ; OR TOGETHER LAST DIGIT AND SIGN
U 1264, 0880,6002,4030,C467,0135,C  ;30510      NEXT/DS.CVTSP.SETCC              ;
                                ;30511
                                ;30512          ;1------------------------------;
U 1265, 0C80,0036,4030,0047,00FF,8  ;30513      NEXT/IE.OPER.FAULT               ;Inst. will pack up to service this
                                ;30514
                                ;30515
                                ;30516 DS.CVTSP.SETCC:
                                ;30517          ;--------------------------------;
U 135C, 0D86,CC37,0030,C047,0135,D  ;30518      M[FPDOFFSET]_ZLIT0[18]           ; LOAD OFFSET OF SPECIAL PACK ROUTINE
                                ;30519
                                ;30520          ;--------------------------------;
U 135D, 0884,0E7D,0034,4047,0135,0  ;30521      R[R1]_RB-1,NEXT/DS.CVTTP.SETCC   ;
                                ;30522
                                ;30523
                                ;30524          ;*********************************************************************
                                ;30525          ; Zero length destination, here after returning from DS.CVTNP on a
                                ;30526          ; FLAG1? BUT (=0* is no OV, =1* is OV).
                                ;30527          ;*********************************************************************
                                ;30528 =0*
                                ;30529 DS.CVTSP.0.DEST:
                                ;30530          ;0*-------------------------------;
                                ;30531          WRITE ZLIT0[0C],SIZE[BYTE],      ;Write +0, wrap up inst.
U 10B8, 0980,0C37,0000,65D8,011E,9  ;30532      NEXT/DS.CVTXX.SETCC.06
                                ;30533
                                ;30534          ;1*-------------------------------;
                                ;30535          WRITE M[TEMP3],SIZE[BYTE],       ;Write sign of dest., ditto
U 10BA, 0880,3592,4000,05D8,011E,9  ;30536      NEXT/DS.CVTXX.SETCC.06
```

```
                                ;30537 .TOC "  Decimal String                    : DS.CVTNP"
                                ;30538
                                ;30539 ;*****************************************************************************
                                ;30540 ;       Entry point     DS.CVTNP
                                ;30541 ;
                                ;30542 ;       Input           TEMP1
                                ;30543 ;
                                ;30544 ;       Resources       TEMP1           Source length
                                ;30545 ;                       TEMP4           Destination length
                                ;30546 ;                       TEMP5           Constant 3030
                                ;30547 ;                       TEMP6           Source digits
                                ;30548 ;                       MTEMP8          Constant F0F0
                                ;30549 ;                       STEPC           Loop on remaining source digits
                                ;30550 ;                       QREG            Temporary
                                ;30551 ;
                                ;30552 ;       Subroutines     DS.CVTXX.SETUP
                                ;30553 ;                       IE.SERV.IP.TS2
                                ;30554 ;                       IE.OPER.FAULT
                                ;30555 ;
                                ;30556 ;       DS.CVTNP is called by DS.CVTTP and DS.CVTSP to do the conversion
                                ;30557 ;       except for the sign and the last digit since these are handled
                                ;30558 ;       differently for trailing numeric and leading seperate strings.
                                ;30559 ;       The source is read and converted a word (2 digits) at a time.
                                ;30560 ;*****************************************************************************
                                ;30561
                                ;30562 DS.CVTNP:
                                ;30563          ;-------------------------------;
U 135E, 0780,07FF,F878,7847,0135,F ;30564       LONLIT_[0F0F0]                  ; MASK TO CHECK FOR VALID ASCII DIGITS
                                ;30565
                                ;30566          ;-------------------------------;
U 135F, 0086,85BE,403D,4047,0126,6 ;30567       M[TEMP8]_R[LONLIT]              ; SAVE MASK
                                ;30568
                                ;30569 =0       ;0------------------------------;
                                ;30570          LONLIT_[3030],                  ; MASK TO CHECK FOR ASCII ZERO
U 1266, 0780,07FF,FE7E,7847,0536,8 ;30571       PUSH,NEXT/DS.CVTXX.SETUP        ;
                                ;30572
                                ;30573          ;1------------------------------;
                                ;30574          STEPC_M[TEMP1]-R[TEMP4],        ; LOAD COUNT OF EXCESS SOURCE DIGITS
U 1267, 0480,1000,0B61,0107,091D,0 374* ;30575  SIZE[LONG],SIGND CMP?            ; COMPARE SOURCE AND DESTINATION LENGTHS
                                ;30576 =00
                                ;30577 DS.CVTNP.STRIP.ZEROS:
                                ;30578          ;00-----------------------------;
                                ;30579          Q_SEXT(XB)-R[TEMP5] PC_PC+1,    ;Throw away leading zeros and check
                                ;30580          SIGND CMP?,                     ;for overflow
U 11D0, 0881,7814,1B41,6047,0909,6 385* ;30581  NEXT/DS.CVTNP.CHK.OVFLW
                                ;30582
                                ;30583          ;01-----------------------------;
                                ;30584          WB_M[TEMP1]-ZLIT0[2],           ; CHECK # OF REMAINING SOURCE DIGITS
                                ;30585          SIGND CMP DEF?,SIZE[LONG],      ;
U 11D1, 0180,1C10,0B60,1047,091D,4 413* ;30586  NEXT/DS.CVTNP.CONVERT            ;
                                ;30587
                                ;30588          ;10-----------------------------; LOAD # OF EXCESS DESTINATION DIGITS
                                ;30589          STEPC_M[TEMP4]-R[TEMP1],        ; TO FILL WITH ZEROS
U 11D2, 0880,4000,0030,4107,0136,0 ;30590       NEXT/DS.CVTNP.50                ;
                                ;30591 =
```

```
                                ;30592 =10
                                ;30593 DS.CVTNP.CHK.OVFLW:
                                ;30594          ;1----------------------------;
                                ;30595          R[TEMP6]_M[TEMP5]+Q,          ;Undo earlier subtraction and see if
U 1096, 0C84,5009,0031,8047,0136,3 ;30596       NEXT/DS.CVTNP.CHK.BAD        ;this is a legal BYTE
                                ;30597
                                ;30598          ;1----------------------------;
                                ;30599          M[TEMP1]_MB-ZLIT0[1],        ;
                                ;30600          DBZ STEPC?,                  ;
U 1097, 0586,1C10,0330,0847,011D,0 ;30601       NEXT/DS.CVTNP.STRIP.ZEROS    ;
                                ;30602
                                ;30603
                                ;30604 DS.CVTNP.50:
                                ;30605          ;----------------------------; CHECK IF ODD OR EVEN
U 1360, 0C80,4592,42B0,0047,0126,8 ;30606       WB_M[TEMP4],WB<0>?           ; LENGTH DESTINATION
                                ;30607
                                ;30608 =0       ;0----------------------------;
                                ;30609          WRITE ZLIT0[0],SIZE[BYTE],   ; WRITE ZEROS TO DESTINATION
                                ;30610          DBZ STEPC?,                  ; DECREMENT COUNT OF DESTINATION DIGITS
U 1268, 0D80,0C37,0300,05D8,0126,C ;30611       NEXT/DS.CVTNP.FILL           ;
                                ;30612
                                ;30613          ;1----------------------------;
U 1269, 0480,0036,4330,0047,0126,A ;30614       DBZ STEPC?                   ; DECREMENT COUNT OF DESTINATION DIGITS
                                ;30615 =0
                                ;30616 DS.CVTNP.STORE.ZERO:
                                ;30617          ;0----------------------------;
                                ;30618          WRITE ZLIT0[0],SIZE[BYTE],   ; WRITE ZEROS TO DESTINATION
                                ;30619          DBZ STEPC?,                  ; DECREMENT COUNT OF DESTINATION DIGITS
U 126A, 0D80,0C37,0300,05D8,0126,C ;30620       NEXT/DS.CVTNP.FILL           ;
                                ;30621
                                ;30622 DS.CVTNP.FILL.DONE:
                                ;30623          ;1----------------------------;
                                ;30624          WB_M[TEMP1]-ZLIT0[2],        ; CHECK # OF SOURCE DIGITS REMAINING
                                ;30625          SIGND CMP DEF?,SIZE[LONG],   ;
U 126B, 0180,1C10,0B60,1047,091D,4 413* ;30626  NEXT/DS.CVTNP.CONVERT         ;
                                ;30627 =0
                                ;30628 DS.CVTNP.FILL:
                                ;30629          ;0----------------------------; MORE TO FILL
                                ;30630          VA_M[VA]+ZLIT0[1],           ; BUMP DESTINATION ADDRESS
                                ;30631          DBZ STEPC?,                  ;
U 126C, 0D81,BC11,0330,0CA7,0126,A ;30632       NEXT/DS.CVTNP.STORE.ZERO     ;
                                ;30633
                                ;30634          ;1----------------------------; FILL COMPLETE
                                ;30635          VA_M[VA]+ZLIT0[1],           ; BUMP DESTINATION ADDRESS
U 126D, 0581,BC11,0030,0CA7,0126,B ;30636       NEXT/DS.CVTNP.FILL.DONE
```

```
                                ;30637 =00
                                ;30638 DS.CVTNP.CONVERT:
                                ;30639          ;00--------------------------------;
                                ;30640          STEPC_M[TEMP1]-ZLIT0[1],        ; LOAD # OF SOURCE BYTES REMAINING - 1
U 11D4, 0580,1C10,02B0,0907,091D,8 377* ;30641  WB<0>?,NEXT/DS.CVTNP.60          ;
                                ;30642
                                ;30643          ;01--------------------------------;
                                ;30644          R[TEMP6]_ZEXT(XB) PC_PC+1,      ; FETCH NEXT TO LAST SOURCE BYTE
U 11D5, 0C85,759E,4001,A047,0136,1      ;30645  NEXT/DS.CVTNP.SINGLE            ;
                                ;30646
                                ;30647          ;10--------------------------------;
U 11D6, 0480,1592,4A30,0047,0926,E 322* ;30648  WB_M[TEMP1],WX.EQ.0?             ;How many digits are left to convert?
                                ;30649                                          ;(0 or 1)
                                ;30650 =
                                ;30651
                                ;30652 =0
                                ;30653 DS.CVTNP.RETURN:
                                ;30654          ;0--------------------------------;
U 126E, 0884,05B7,00B4,8047,0000,1      ;30655  R[R2]_0,RETURN [+1]             ;1 digit left, normal return
                                ;30656
                                ;30657          ;1--------------------------------;
                                ;30658          R[R2]_0,MDR_0,SET FLAG2,        ;No digits left(0 len. dest.),
U 126F, 0854,05B7,00B4,84E7,0000,2      ;30659  RETURN [+2]                     ;<Z>image_1, take special RETURN
                                ;30660
                                ;30661 =00
                                ;30662 DS.CVTNP.60:
                                ;30663          ;00--------------------------------; ODD # OF SOURCE DIGITS TO CONVERT
                                ;30664          R[TEMP6]_ZEXT(XB) PC_PC+2,      ; FETCH NEXT 2 SOURCE DIGITS
U 11D8, 009D,759E,4011,A047,0127,0      ;30665  DEC STEPC,NEXT/DS.CVTNP.LOOP    ; DECREMENT # OF REMAINING SOURCE DIGITS
                                ;30666
                                ;30667          ;01--------------------------------; EVEN # OF SOURCE DIGITS TO CONVERT
                                ;30668          R[TEMP6]_ZEXT(XB) PC_PC+1,      ; FETCH NEXT SOURCE DIGIT
                                ;30669          DEC STEPC,                      ; DECREMENT # OF REMAINING SOURCE DIGITS
U 11D9, 049D,759E,4001,A047,0536,1      ;30670  PUSH,NEXT/DS.CVTNP.SINGLE       ; GO WRITE FIRST DESTINATION BYTE
                                ;30671
                                ;30672          ;10--------------------------------;
                                ;30673          R[TEMP6]_ZEXT(XB) PC_PC+2,      ; FETCH NEXT 2 SOURCE DIGITS
U 11DA, 009D,759E,4011,A047,0127,0      ;30674  DEC STEPC,NEXT/DS.CVTNP.LOOP    ; DECREMENT # OF REMAINING SOURCE DIGITS
                                ;30675 =
                                ;30676 =0
                                ;30677 DS.CVTNP.LOOP:
                                ;30678          ;0--------------------------------;
                                ;30679          Q_M[TEMP8].AND.R[TEMP6],        ; MASK TO CHECK FOR LEGAL ASCII DIGITS
                                ;30680          F[AG2?,                         ; CHECK FOR ZERO UNTIL FIRST
U 1270, 0C80,8002,14B1,8047,0107,1      ;30681  NEXT/DS.CVTNP.CHK.INPUT         ; NON ZERO DIGIT FOUND
                                ;30682
                                ;30683          ;1--------------------------------;
                                ;30684          INTPEND OR TIMER?,              ;
U 1271, 0480,0036,4AF0,0047,04F7,0      ;30685  PUSH,NEXT/IE.SERV.IP.TS2        ;
```

```
                                        ;30686 =0*0
                                        ;30687 DS.CVTNP.NON.ZERO:
                                        ;30688          ;0*0-------------------------------;
                                        ;30689          WB_M[TEMP5].XOR.Q,              ; CHECK FOR LEGAL ASCII DIGITS
                                        ;30690          CLEAR FLAG2,                    ; CLEAR PSL<Z>
U 1070, 0010,500B,4A30,0047,0127,2      ;30691          WX.EQ.0?,NEXT/DS.CVTNP.80       ;
                                        ;30692
                                        ;30693 DS.CVTNP.CHK.INPUT:
                                        ;30694          ;0*1-------------------------------;
                                        ;30695          WB_M[TEMP5].XOR.Q,              ; CHECK FOR LEGAL ASCII DIGITS
U 1071, 0080,500B,4A30,0047,0127,2      ;30696          WX.EQ.0?,NEXT/DS.CVTNP.80       ;
                                        ;30697
                                        ;30698 =1*1     ;1*1-------------------------------;
                                        ;30699          WB_M[TEMP5].XOR.R[TEMP6],       ; CHECK FOR ASCII ZERO
U 1075, 0480,5003,4A31,8047,0107,0      ;30700          WX.EQ.0?,NEXT/DS.CVTNP.NON.ZERO ;
                                        ;30701 =0
                                        ;30702 DS.CVTNP.80:
                                        ;30703          ;0-------------------------------;
U 1272, 0C80,0036,4030,0047,00FF,8      ;30704          NEXT/IE.OPER.FAULT              ;Inst. will pack up to service this
                                        ;30705                                          ;fault
                                        ;30706
                                        ;30707          ;1-------------------------------;
                                        ;30708          WRITE CVTNP(M[TEMP6]),          ;
                                        ;30709          SIZE[BYTE],                     ; WRITE 2 DESTINATION DIGITS (1 BYTE)
U 1273, 0880,67FD,498D,85D8,0927,4 387* ;30710          BCD CHECK?                      ; FINISH CHECK OF INPUT DATA
                                        ;30711
                                        ;30712 =0       ;0-------------------------------;
                                        ;30713          VA_M[VA]+ZLIT0[1],              ; INCREMENT DESTINATION ADDRESS
U 1274, 0581,BC11,0330,0CA7,0127,6      ;30714          DBZ STEPC?,NEXT/DS.CVTNP.90     ; DECREMENT # OF REMAINING SOURCE DIGITS
                                        ;30715
                                        ;30716          ;1-------------------------------;
U 1275, 0C80,0036,4030,0047,00FF,8      ;30717          NEXT/IE.OPER.FAULT              ;Inst. will pack up to service this
                                        ;30718                                          ;fault
                                        ;30719
                                        ;30720 =0
                                        ;30721 DS.CVTNP.90:
                                        ;30722          ;0-------------------------------;
                                        ;30723          R[TEMP6]_ZEXT(XB) PC_PC+2,      ; GET 2 SOURCE DIGITS
                                        ;30724          DEC STEPC,                      ; DECREMENT # OF REMAINING SOURCE DIGITS
U 1276, 049D,759E,4B11,B8E7,0127,0      ;30725          IP.TS?,NEXT/DS.CVTNP.LOOP       ; CHECK FOR INTERRUPTS AND LOOP
                                        ;30726
                                        ;30727          ;1-------------------------------;
U 1277, 0884,05B7,00B4,8047,0000,1      ;30728          R[R2]_0,RETURN [+1]             ; 1 SOURCE DIGIT REMAINS, NORMAL RETURN
```

```
                                ;30729 DS.CVTNP.SINGLE:
                                ;30730          ;--------------------------------;
U 1361, 0480,8002,2031,8047,0136,2 ;30731       D_M[TEMP8].AND.R[TEMP6]          ; MASK TO CHECK FOR 0 AND VALID DATA
                                ;30732
                                ;30733          ;--------------------------------;
                                ;30734          WB_D.XOR.ZLIT0[30],              ; CHECK FOR VALID ASCII DIGIT
U 1362, 0D80,0C33,4A31,8047,0127,8 ;30735       WX.EQ.0?                         ;
                                ;30736
                                ;30737 =0       ;0------------------------------;
U 1278, 0C80,0036,4030,0047,00FF,8 ;30738       NEXT/IE.OPER.FAULT               ;Pack up to service this fault
                                ;30739
                                ;30740          ;1------------------------------;
                                ;30741          M[TEMP6]_CVTNP(MB),              ; CONVERT SINGLE DIGIT
U 1279, 0086,67FD,49BD,8047,0927,A 387* ;30742  BCD CHECK?                        ; FINISH CHECK FOR VALID ASCII DIGIT
                                ;30743
                                ;30744 =0       ;0------------------------------;
                                ;30745          WRITE (M[TEMP6] R[TEMP6]).RR.4,  ; WRITE FIRST DIGIT AND LEADING 0 NIBBLE
                                ;30746          SIZE[BYTE],WX.EQ.0?,             ; CHECK FOR ZERO FOR SETTING CC
U 127A, 0C80,6277,0A01,85D8,0127,C ;30747       NEXT/DS.CVTNP.SINGLE.10          ;
                                ;30748
                                ;30749          ;1------------------------------;
U 1278, 0C80,0036,4030,0047,00FF,8 ;30750       NEXT/IE.OPER.FAULT               ;Pack up to service this fault
                                ;30751
                                ;30752 =0
                                ;30753 DS.CVTNP.SINGLE.10:
                                ;30754          ;0------------------------------;
                                ;30755          VA_M[VA]+ZLIT0[1],CLEAR FLAG2,   ; INCREMENT DESTINATION ADDRESS
U 127C, 0511,BC11,0030,0CA7,0126,E ;30756       NEXT/DS.CVTNP.RETURN             ; CLEAR PSL<Z> AND TAKE NORMAL RETURN
                                ;30757
                                ;30758          ;1------------------------------;
                                ;30759          VA_M[VA]+ZLIT0[1],               ; INCREMENT DESTINATION ADDRESS
U 127D, 0581,BC11,0030,0CA7,0126,E ;30760       NEXT/DS.CVTNP.RETURN             ; TAKE NORMAL RETURN
```

```
                                        ;30761          ;*********************************************************************
                                        ;30762          ; Here when leading BYTE of source isn't ASCII zero.  If illegal
                                        ;30763          ; digit, take a reserved operand fault, else set <V>image.
                                        ;30764          ;*********************************************************************
                                        ;30765
                                        ;30766 DS.CVTNP.CHK.BAD:
                                        ;30767          ;-------------------------------;
U 1363, 0880,67FD,49BD,8047,08BF,4 387* ;30768          BCD CHECK M[TEMP6]?
                                        ;30769
                                        ;30770 0BF4:    ;0********FORCE ADDRESS*********;
                                        ;30771          D_M[TEMP8].AND.R[TEMP6],        ; CHECK THAT BITS<7:4> ARE 3 FOR ASCII
U 0BF4, 0C80,8002,2031,8047,00BF,3      ;30772          NEXT/DS.CVTNP.CHK.10            ; 0-9 (HEX 30-39)
                                        ;30773
                                        ;30774 0BF5:    ;1********FORCE ADDRESS*********;
U 0BF5, 0C80,0036,4030,0047,00FF,8      ;30775          NEXT/IE.OPER.FAULT              ; BITS<3:0> ARE GREATER THAN 9
                                        ;30776
                                        ;30777 0BF3:
                                        ;30778 DS.CVTNP.CHK.10:
                                        ;30779          ;*********FORCE ADDRESS*********;
                                        ;30780          WB_D.XOR.ZLIT0[30],             ;
U 0BF3, 0980,0C33,4A71,8047,0127,E      ;30781          WX.NE.0?
                                        ;30782
                                        ;30783 =0
                                        ;30784          ;0-------------------------------; OVERFLOW, SET V FLAG
                                        ;30785          M[TEMP1]_MB-ZLIT0[1],SET FLAG1,
                                        ;30786          DBZ STEP[?,
U 127E, 0D4E,1C10,0330,0847,011D,0      ;30787          NEXT/DS.CVTNP.STRIP.ZEROS
                                        ;30788
                                        ;30789          ;1-------------------------------;
U 127F, 0C80,0036,4030,0047,00FF,8      ;30790          NEXT/IE.OPER.FAULT
```

```
                                ;30791 .TOC "  Decimal String                    : DS.CVTPN"
                                ;30792
                                ;30793 ;*****************************************************************************
                                ;30794 ;       Entry point     DS.CVTPN
                                ;30795 ;
                                ;30796 ;       Input           TEMP1
                                ;30797 ;
                                ;30798 ;       Resources       TEMP0           Constant 33
                                ;30799 ;                       TEMP1           Zext(source length)
                                ;30800 ;                       TEMP4           Zero extended destination length
                                ;30801 ;                       TEMP5           Constant 30303030
                                ;30802 ;                       TEMP6           Source byte
                                ;30803 ;                       STEPC           Loop control
                                ;30804 ;
                                ;30805 ;       Output          MEMORY          All but last digit and sign written
                                ;30806 ;                       FLAGS           Intermediate condition code setting
                                ;30807 ;                       TEMP6           Last source byte (digit and sign)
                                ;30808 ;
                                ;30809 ;       Subroutines     DS.CVTXX.SETUP
                                ;30810 ;                       IE.OPER.FAULT
                                ;30811 ;                       DS.INC.VA.BY.2
                                ;30812 ;                       IE.SERV.IP.TS2
                                ;30813 ;
                                ;30814 ;       DS.CVTPN is used to do most of the conversion for DS.CVTPT and DS.CVTPS.
                                ;30815 ;       It converts all but the last digit and sign which are handled
                                ;30816 ;       differently for trailing numeric and leading seperate strings.
                                ;30817 ;*****************************************************************************
                                ;30818
                                ;30819 =0
                                ;30820 DS.CVTPN:
                                ;30821          ;0-------------------------------;
                                ;30822          LONLIT [30303030],              ; USED TO WRITE LEADING ZEROS
1280, 0780,067E,7E7F,7847,0536,B ;30823         PUSH,NEXT/DS.CVTXX.SETUP        ;
                                ;30824
                                ;30825          ;1-------------------------------;
1281, 0086,027E,4031,4047,0136,4 ;30826         M[TEMP0]_R[TEMP5].OR.((MB RB).RR.4) ; USED FOR CVTPN ROTATER FUNCTION
                                ;30827
                                ;30828          ;-------------------------------;
                                ;30829          D_M[TEMP1]-R[TEMP4],            ; LOAD # OF EXCESS SOURCE DIGITS
1364, 0080,1000,2861,0047,091D,0 374*  ;30830   SIGND CMP?,SIZE[LONG]            ; COMPARE SOURCE AND DESTINATION LENGTHS
```

```
                                    ;30831 =00      ;00------------------------------; SOURCE LENGTH IS GREATER
                                    ;30832          WB_M[TEMP1],WB<0>?,              ; ODD OR EVEN SOURCE LENGTH? THROW AWAY
U 11DC, 0480,1592,42B0,0047,0128,E  ;30833          NEXT/DS.CVTPN.STRIP.ZEROS        ; EXTRA DIGITS CHECKING FOR OVERFLOW
                                    ;30834
                                    ;30835          ;01------------------------------; LENGTHS ARE THE SAME
                                    ;30836          WB_M[TEMP1],WX.EQ.0?,            ; ODD OR EVEN SOURCE LENGTH?
U 11DD, 0C80,1592,4A30,0047,0928,4 322* ;30837      NEXT/DS.CVTPN.CONVERT            ; LENGTHS ARE THE SAME, DO THE CONVERT
                                    ;30838
                                    ;30839          ;10------------------------------; DESTINATION LENGTH IS GREATER
U 11DE, 0880,4000,0030,4107,0136,5  ;30840          STEPC_M[TEMP4]-R[TEMP1]          ; LOAD # OF LEADING ZEROS TO WRITE
                                    ;30841 =
                                    ;30842
                                    ;30843          ;--------------------------------;
U 1365, 0880,0036,4370,0047,010F,0  ;30844          STEPC.GE.4? DECBY4               ; ARE THERE 4 OR MORE ZEROS TO WRITE?
                                    ;30845 =000
                                    ;30846 DS.CVTPN.WRT.ZEROS:
                                    ;30847          ;000-----------------------------;
                                    ;30848          WB_M[TEMP1],WX.EQ.0?,            ; CHECK FOR ZERO LENGTH SOURCE
U 10F0, 0C80,1592,4A30,0047,0928,2 322* ;30849      NEXT/DS.CVTPN.40                 ;
                                    ;30850
                                    ;30851          ;001-----------------------------;
                                    ;30852          WRITE M[TEMP5],SIZE[BYTE],       ; WRITE LAST LEADING ZERO
U 10F1, 0080,5592,4000,05D8,010F,7  ;30853          NEXT/DS.CVTPN.20                 ;
                                    ;30854
                                    ;30855          ;010-----------------------------;
                                    ;30856          WRITE M[TEMP5],SIZE[WORD],       ; WRITE LAST 2 LEADING ZEROS
U 10F2, 0C80,5592,4010,05D8,0538,0  ;30857          PUSH,NEXT/DS.INC.VA.BY.2         ; INCREMENT DESTINATION ADDRESS BY 2
                                    ;30858
                                    ;30859          ;011-----------------------------;
                                    ;30860          WRITE M[TEMP5],SIZE[WORD],       ; WRITE 2 ZEROS THEN ONE ZERO
U 10F3, 0C80,5592,4010,05D8,0538,0  ;30861          PUSH,NEXT/DS.INC.VA.BY.2         ; INCREMENT DESTINATION ADDRESS BY 2
                                    ;30862
                                    ;30863          ;100-----------------------------;
                                    ;30864          WRITE M[TEMP5],SIZE[LONG],       ; WRITE 4 ZEROS AND LOOP
U 10F4, 0C80,5592,4020,05D8,010F,6  ;30865          NEXT/DS.CVTPN.WRT.ZERO.LP        ;
                                    ;30866
                                    ;30867          ;101-----------------------------;
                                    ;30868          INTPEND OR TIMER?,               ;
U 10F5, 0480,0036,4AF0,0047,04F7,0  ;30869          PUSH,NEXT/IE.SERV.IP.TS2         ;
                                    ;30870
                                    ;30871 DS.CVTPN.WRT.ZERO.LP:
                                    ;30872          ;--------------------------------;
                                    ;30873          VA_M[VA]+ZLIT0[4],               ; INCREMENT DESTINATION ADDRESS BY 4
                                    ;30874          STEPC.GE.4? DECBY4,
U 10F6, 0D81,BC11,0370,24A7,010F,0  ;30875          NEXT/DS.CVTPN.WRT.ZEROS
```

```
                                ;30876 DS.CVTPN.20:
                                ;30877          ;-------------------------------;
                                ;30878          VA_M[VA]+ZLIT0[1],              ; INCREMENT DESTINATION ADDRESS BY 1
U 10F7, 0581,BC11,0030,0CA7,010F,0  ;30879      NEXT/DS.CVTPN.WRT.ZEROS         ;
                                ;30880 =0
                                ;30881 DS.CVTPN.40:
                                ;30882          ;0-------------------------------; LENGTHS ARE THE SAME, DO THE CONVERT
                                ;30883          STEPC_M[TEMP1]-R[ZERO],         ; LOAD # OF REMAINING SOURCE DIGITS
U 1282, 0480,1000,02BD,8107,0928,6 344*  ;30884 WB<0>?,NEXT/DS.CVTPN.EVEN.LEN   ; CHECK IF ODD OR EVEN LENGTH
                                ;30885
                                ;30886          ;1-------------------------------; ZERO LENGTH SOURCE, POINT VA TO
                                ;30887          VA_M[VA]-ZLIT0[1],SET FLAG2,    ; LAST DESTINATION BYTE, NORMAL RETURN
U 1283, 0151,BC10,0030,0CA7,010F,8  ;30888      NEXT/DS.CVTPN.RETURN            ;
                                ;30889 =0
                                ;30890 DS.CVTPN.CONVERT:
                                ;30891          ;0-------------------------------; LENGTHS ARE THE SAME, DO THE CONVERT
                                ;30892          STEPC_M[TEMP1]-R[ZERO],         ; LOAD # OF REMAINING SOURCE DIGITS
U 1284, 0480,1000,02BD,8107,0928,6 344*  ;30893 WB<0>?,NEXT/DS.CVTPN.EVEN.LEN   ; CHECK IF ODD OR EVEN LENGTH
                                ;30894
                                ;30895          ;1-------------------------------; ZERO LENGTH DESTINATION,
                                ;30896          R[TEMP6]_ZEXT(XB) PC_PC+1,      ; GET LAST SOURCE BYTE
                                ;30897          ALUS_BCD SIGN.ZERO,             ; LATCH SIGN FROM LAST BYTE
U 1285, 0885,759E,4081,A0C7,0000,2  ;30898      RETURN [+2]                     ; TAKE SPECIAL RETURN
                                ;30899 =0
                                ;30900 DS.CVTPN.EVEN.LEN:
                                ;30901          ;0-------------------------------; EVEN LENGTH, CONVERT 1 DIGIT FIRST.
                                ;30902          R[TEMP6]_ZEXT(XB) PC_PC+1,      ; GET SOURCE BYTE, CHECK FOR ZERO AND
                                ;30903          DEC STEPC,                      ; DECREMENT # OF SOURCE DIGITS REMAINING
U 1286, 049D,759E,4A01,A047,0928,8 322*  ;30904 WX.EQ.0?,NEXT/DS.CVTPN.SINGLE   ;
                                ;30905
                                ;30906 DS.CVTPN.ODD.LEN:
                                ;30907          ;1-------------------------------; ODD LENGTH, CONVERT 4 AT A TIME
                                ;30908          VA_M[VA]-ZLIT0[4],              ; BACKUP DEST ADDRESS TO BEGIN LOOP
U 1287, 0599,BC10,0030,24A7,0136,8  ;30909      DEC STEPC,NEXT/DS.CVTPN.LOOP    ; DECREMENT # OF SOURCE DIGITS REMAINING
                                ;30910 =0
                                ;30911 DS.CVTPN.SINGLE:
                                ;30912          ;0-------------------------------;
                                ;30913          M[TEMP6]_(R[TEMP6] MB).RL.4,    ; SHIFT TO CONVERT ONLY 1 PACKED DIGIT
                                ;30914          CLEAR FLAG2,                    ; CLEAR PSL<Z> SINCE DIGIT IS NON ZERO.
U 1288, 0816,6237,0031,8047,0136,6  ;30915      NEXT/DS.CVTPN.SINGLE.WRT        ;
                                ;30916
                                ;30917          ;1-------------------------------;
                                ;30918          M[TEMP6]_(R[TEMP6] MB).RL.4,    ; SHIFT TO CONVERT ONLY 1 PACKED DIGIT
U 1289, 0886,6237,0031,8047,0136,6  ;30919      NEXT/DS.CVTPN.SINGLE.WRT        ;
                                ;30920
                                ;30921 DS.CVTPN.SINGLE.WRT:
                                ;30922          ;-------------------------------;
                                ;30923          WRITE CVTPN(M[TEMP6]),          ;
U 1366, 0498,66F7,0000,05D8,0136,7  ;30924      SIZE[BYTE],DEC STEPC            ; WRITE FIRST DESTINATION DIGIT
```

```
                                ;30925          ;--------------------------------; INCREMENT DESTINATION ADDRESS.
U 1367, 0D81,BC10,0030,1CA7,0136,8  ;30926          VA_M[VA]-ZLIT0[3]               ; (VA-3) + 4(IN NEXT CYCLE) = VA + 1
                                ;30927
                                ;30928 DS.CVTPN.LOOP:
                                ;30929          ;--------------------------------;
                                ;30930          VA_VA+4,                        ; INC DESTINATION ADDRESS.
U 1368, 0880,0036,4370,0447,010F,8  ;30931          STEPC.GE.4? DECBY4              ; ARE THERE 5 OR MORE DIGITS REMAINING.
                                ;30932 ;=000
                                ;30933 10F8:
                                ;30934 DS.CVTPN.RETURN:
                                ;30935          ;0C0-----------------------------; 1 DIGIT REMAINING.
                                ;30936          R[TEMP6]_ZEXT(XB) PC_PC+1,      ; GET LAST SOURCE BYTE.
                                ;30937          ALUS_BCD SIGN.ZERO,             ; LATCH BCD SIGN.
U 10F8, 088 ,759E,4081,A0C7,0000,1  ;30938          RETURN [+1]                     ;
                                ;30939 10FA:
                                ;30940 ;=010    ;010-----------------------------; 3 DIGITS REMAINING.
                                ;30941          R[TEMP6]_ZEXT(XB) PC_PC+1,      ; GET NEXT TO LAST BYTE.
U 10FA, 0C85,759E,4A01,A047,0928,A 322*  ;30942          WX.EQ.0?,NEXT/DS.CVTPN.LAST3    ; CHECK FOR NON ZERO DIGITS.
                                ;30943
                                ;30944 10FC:
                                ;30945 ;=100    ;100-----------------------------; 5 OR MORE DIGITS REMAINING.
                                ;30946          R[TEMP6]_ZEXT(XB) PC_PC+2,      ; GET NEXT 4 DIGITS (2 BYTES)
                                ;30947          ALUS_BCD SIGN.ZERO,             ; LATCH BCD SIGN (IN CASE LAST).
U 10FC, 0085,759E,4A11,A0C7,0928,C 322*  ;30948          WX.EQ.0?,NEXT/DS.CVTPN.WRT.LONG ;
                                ;30949
                                ;30950 10FD:    ;101-----------------------------; 5 OR MORE DIGITS AND AN INTERRUPT
                                ;30951          INTPEND OR TIMER?,              ; IS PENDING.
U 10FD, 0480,0036,4AF0,0047,04F7,0  ;30952          PUSH,NEXT/IE.SERV.IP.TS2        ;
                                ;30953 ;=
                                ;30954 =0
                                ;30955 DS.CVTPN.LAST3:
                                ;30956          ;0--------------------------------;
                                ;30957          WRITE CVTPN(M[TEMP6]),          ; CONVERT AND WRITE 2 DIGITS.
                                ;30958          SIZE[WORD],CLEAR FLAG2,         ; CLEAR PSL Z (NON ZERO DIGITS)
U 128A, 0810,66F7,0010,05D8,0136,9  ;30959          NEXT/DS.CVTPN.INC.VA            ;
                                ;30960
                                ;30961          ;1--------------------------------;
                                ;30962          WRITE CVTPN(M[TEMP6]),          ; CONVERT AND WRITE 2 DIGITS.
U 128B, 0880,66F7,0010,05D8,0136,9  ;30963          SIZE[WORD]                      ;
                                ;30964
                                ;30965 DS.CVTPN.INC.VA:
                                ;30966          ;--------------------------------;
                                ;30967          VA_M[VA]+CONX(2),               ; INCREMENT DESTINATION ADDRESS.
U 1369, 0081,B711,0010,04A7,010F,8  ;30968          NEXT/DS.CVTPN.RETURN            ;
                                ;30969 =0
                                ;30970 DS.CVTPN.WRT.LONG:
                                ;30971          ;--------------------------------;
                                ;30972          WRITE CVTPN(M[TEMP6]),          ; CONVERT AND WRITE 4 DIGITS.
                                ;30973          SIZE[LONG],CLEAR FLAG2,         ; CLEAR PSL Z (NON ZERO DIGITS)
U 128C, 0010,66F7,0020,05D8,0136,8  ;30974          NEXT/DS.CVTPN.LOOP              ;
                                ;30975
                                ;30976          ;1--------------------------------;
                                ;30977          WRITE CVTPN(M[TEMP6]),          ; CONVERT AND WRITE 4 DIGITS.
U 128D, 0080,66F7,0020,05D8,0136,8  ;30978          SIZE[LONG],NEXT/DS.CVTPN.LOOP   ;
```

```
                                ;30979 =0
                                ;30980 DS.CVTPN.STRIP.ZEROS:
                                ;30981          ;0------------------------------; SOURCE LENGTH IS EVEN,
                                ;30982          R[TEMP6]_ZEXT(XB) PC_PC+1,      ;
                                ;30983          ALUS_BCD SIGN.ZERO,WX.EQ.0?,    ; STRIP 1 LEADING ZERO AND CHECK
U 128E, 0485,759E,4A01,A0C7,0929,2 322*  ;30984          NEXT7DS.CVTPN.STRIP.SNGLE       ; FOR OVERFLOW
                                ;30985
                                ;30986          ;1------------------------------; WHEN D BECOMES NEGATIVE ALL
U 128F, 0180,0C30,2030,0847,0136,A       ;30987          D_D-ZLIT0[1]                    ; LEADING DIGITS HAVE BEEN STRIPED.
                                ;30988
                                ;30989          ;------------------------------;
                                ;30990          R[TEMP6]_ZEXT(XB) PC_PC+1,      ;
                                ;30991          ALUS_BCD SIGN.ZERO,WX.EQ.0?,    ; CHECK FIRST 2 DIGITS FOR OVERFLOW
U 136A, 0C85,759E,4A01,A0C7,0929,0 322*  ;30992          NEXT7DS.CVTPN.STRIP.LOOP        ;
                                ;30993
                                ;30994 =0
                                ;30995 DS.CVTPN.STRIP.LOOP:
                                ;30996          ;0------------------------------;
                                ;30997          D_D-ZLIT0[1],WB<31-30>?,        ;Possible overflow, strip next digit,
U 1290, 0180,0C30,26F0,0847,091E,1 377*  ;30998          NEXT/DS.CVTPN.STRIP.OVFLW       ;check for last source digit
                                ;30999
                                ;31000          ;1------------------------------;
                                ;31001          D_D-ZLIT0[2],WB<31-30>?,        ;Strip next 2 digits, check for last
U 1291, 0980,0C30,26F0,1047,091E,5 377*  ;31002          NEXT/DS.CVTPN.STRIP.MORE        ;source digit
                                ;31003
                                ;31004 =01
                                ;31005 DS.CVTPN.STRIP.OVFLW:
                                ;31006          ;01-----------------------------;
                                ;31007          D_D-ZLIT0[1],WB<31-30>?,        ;Overflow, <V>image_1, strip next
                                ;31008          SET FLAG1,                      ;digit, check for last source digit
U 11E1, 0148,0C30,26F0,0847,091E,5 377*  ;31009          NEXT/DS.CVTPN.STRIP.MORE
                                ;31010
                                ;31011          ;11-----------------------------;
                                ;31012          WB_M[TEMP6].AND.ZLIT0[0F0],     ; CHECK LAST SOURCE DIGIT FOR ZERO
                                ;31013          WX.EQ.0?,                       ;
U 11E3, 0180,6C12,0A37,8047,011E,6       ;31014          NEXT/DS.CVTPN.STRIP.LAST        ;
                                ;31015
                                ;31016 =0
                                ;31017 DS.CVTPN.STRIP.SNGLE:
                                ;31018          ;0------------------------------;
                                ;31019          D_D-ZLIT0[2],WB<31-30>?,        ;Overflow, <V>image_1, strip next digit,
                                ;31020          SET FLAG1,                      ;(dec. DREG by 2, then minus indicates
U 1292, 0148,0C30,26F0,1047,091E,5 377*  ;31021          NEXT/DS.CVTPN.STRIP.MORE        ;done, not zero) check for last source
                                ;31022                                          ;digit
                                ;31023
                                ;31024          ;1------------------------------;
                                ;31025          D_D-ZLIT0[2],WB<31-30>:,        ;Strip next digit, (dec. DREG by 2, then
U 1293, 0980,0C30,26F0,1047,091E,5 377*  ;31026          NEXT/DS.CVTPN.STRIP.MORE        ;minus indicates done, not zero) check
                                ;31027                                          ;for last source digit
```

```
                                ;31028 =00
                                ;31029 =01
                                ;31030 DS.CVTPN.STRIP.MORE:
                                ;31031          ;01-----------------------------; MORE DIGITS TO STRIP
                                ;31032          R[TEMP6]_ZEXT(XB) PC_PC+1,      ; GET NEXT 2 SOURCE DIGITS
                                ;31033          ALUS_BCD SIGN.ZERO,WX.EQ.0?,    ; CHECK FOR OVERFLOW
U 11E5, 0C85,759E,4A01,A0C7,0929,0 322*  ;31034          NEXT/DS.CVTPN.STRIP.LOOP        ; AND LOOP
                                ;31035
                                ;31036 DS.CVTPN.STRIP.LAST:
                                ;31037          ;10-----------------------------; ALL LEADING DIGITS HAVE BEEN STRIPPED
                                ;31038          SET FLAG1,                      ; LAST DIGIT STRIPPED IS OVERFLOW
                                ;31039          M[TEMP1]_R[TEMP4],WX.EQ.0?,     ; SET SOURCE LENGTH EQUAL TO DESTINATION
U 11E6, 0C4E,15BE,4A31,0047,0929,4 322*  ;31040          NEXT/DS.CVTPN.CONVERT.10        ; LENGTH, AND CHECK FOR LENGTH EQUAL 0
                                ;31041
                                ;31042          ;11-----------------------------; ALL LEADING DIGITS HAVE BEEN STRIPPED
                                ;31043          M[TEMP1]_R[TEMP4],WX.EQ.0?,     ; SET SOURCE LENGTH EQUAL TO DESTINATION
U 11E7, 0486,15BE,4A31,0047,0929,4 322*  ;31044          NEXT/DS.CVTPN.CONVERT.10        ; LENGTH, AND CHECK FOR LENGTH EQUAL 0
                                ;31045 =0
                                ;31046 DS.CVTPN.CONVERT.10:
                                ;31047          ;0-----------------------------;
                                ;31048          STEPC_M[TEMP1]-ZLIT0[1],        ;STEPC_#remaining source digits - 1,
U 1294, 0580,1C10,02B0,0907,08BF,8 377*  ;31049          WB<0>?,NEXT/DS.CVTPN.CONVERT.20 ;is result even or odd?
                                ;31050
                                ;31051          ;1-----------------------------; ZERO LENGTH DESTINATION,
U 1295, 0880,0036,40B0,0047,0000,2       ;31052          RETURN [+2]                     ; TAKE SPECIAL RETURN
                                ;31053
                                ;31054 0BF8:
                                ;31055 DS.CVTPN.CONVERT.20:
                                ;31056          ;0********FORCE ADDRESS*********;
                                ;31057          STEPC.GE.4? DECBY4,             ;STEPC is now even, #remaining digits
U 0BF8, 0080,0036,4370,0047,010F,8       ;31058          NEXT/DS.CVTPN.RETURN            ;was odd, convert 4 at a time & begin
                                ;31059
                                ;31060 0BF9:    ;1********FORCE ADDRESS*********;STEPC IS NOW ODD, #REMAINING DIGITS WAS
                                ;31061          WB_M[TEMP6].AND.ZLIT0[0F],      ;EVEN, CHECK DIGIT FOR ZERO THEN
U 0BF9, 0580,6C12,0A30,7847,0128,8       ;31062          WX.EQ.0?,NEXT/DS.CVTPN.SINGLE   ;GO CONVERT IT
                                ;31063
                                ;31064 1297:    ;++++++++ force address ++++++++;
                                ;31065 QU.RESV.OPER:                            ; CAN'T COMPLETE INTERLOCKED QUE INSTR
                                ;31066
                                ;31067          ;-----------------------------; SAVE FAULTING ADDRESS INCASE
U 1297, 0486,65BE,4039,C047,0129,6       ;31068          M[TEMP6]_R[MM.TEMP3]            ; IN CASE OF A TB-MISS ON HEADER
                                ;31069
                                ;31070 1296:    ;+++++++ force address +++++++++;
                                ;31071          ;-----------------------------; GET SET TO READ HEADER
                                ;31072          READ.MOD.LOCK,SIZE[LONG],       ; WITH HARDWARE INTERLOCK IN EFFECT
                                ;31073          CLEAR FPD,                      ;
U 1296, 00E0,0036,4020,0053,0105,2       ;31074          NEXT/QU.RESV.FIX                ;REMAINDER IN DECMAL MODULE
```

```
                                ;31075 .TOC "  Decimal String                    : DS.CVTXX.SETUP"
                                ;31076
                                ;31077 ;*****************************************************************************
                                ;31078 ;       Entry point     DS.CVTXX.SETUP
                                ;31079 ;
                                ;31080 ;       Input           LONLIT          Constant to be loaded into TEMP5
                                ;31081 ;                       TEMP1           Source length
                                ;31082 ;                       TEMP2           Source address
                                ;31083 ;                       TEMP4           Destination length
                                ;31084 ;                       PC              Address of next opcode
                                ;31085 ;                       MDR             Destination address
                                ;31086 ;
                                ;31087 ;       Resources       DREG            ORing the operand lengths
                                ;31088 ;
                                ;31089 ;       Output          R0              Source length
                                ;31090 ;                       R1              Source address
                                ;31091 ;                       R3              Destination address
                                ;31092 ;                       TEMP5           Constant from LONLIT
                                ;31093 ;                       PC              Source address
                                ;31094 ;                       VA              Destination address
                                ;31095 ;                       FLAG2           Set (assume PSL<Z> is set until non
                                ;31096 ;                                       zero found)
                                ;31097 ;
                                ;31098 ;       Subroutines     IE.SERV.IP.TS2
                                ;31099 ;                       IE.OPER.FAULT
                                ;31100 ;
                                ;31101 ;       This setup is called by DS.CVTNP and DS.CVTPN to initialize the
                                ;31102 ;       GPR's and setup PC for sourcing and VA for writing the destination.
                                ;31103 ;
                                ;31104 ;*****************************************************************************
                                ;31105
                                ;31106 DS.CVTXX.SETUP:
                                ;31107          ;--------------------------------;
U 136B, 0C80,1002,6031,0047,0136,C  ;31108      D_M[TEMP1].OR.R[TEMP4]          ;First are operand lengths legal?
                                ;31109
                                ;31110          ;--------------------------------;
                                ;31111          WB_D.ANDNOT.ZLIT0[01F],
U 136C, 0D80,0C33,8A30,F847,0129,8  ;31112      WX.EQ.0?
                                ;31113
                                ;31114 =0
                                ;31115          ;0-------------------------------;
U 1298, 0CE0,0036,4030,0047,00FF,8  ;31116      CLEAR FPD,NEXT/IE.OPER.FAULT    ;No, roll back
                                ;31117
                                ;31118          ;1-------------------------------;
U 1299, 0C54,1592,4034,0047,0136,D  ;31119      R[R0]_M[TEMP1],SET FLAG2        ;Yes, save source length
                                ;31120                                          ; ASSUME PSL<Z> SET UNTIL NON ZERO FOUND
                                ;31121
                                ;31122          ;--------------------------------;
U 136D, 0485,A592,4032,0047,0136,E  ;31123      R[TEMP8]_M[PC]                  ;
                                ;31124
                                ;31125          ;--------------------------------;
U 136E, 0084,2592,4034,4487,0136,F  ;31126      PC_R[R1]_M[TEMP2]               ; POINT PC TO SOURCE
```

```
                                ;31127          ;--------------------------------;
                                ;31128          M[TEMP5]_R[LONLIT],              ;
U 136F, 0086,55BE,4B3D,58E7,0129,A ;31129       IP.TS?                           ; CHECK FOR INTERRUPTS
                                ;31130
                                ;31131 =0
                                ;31132          ;0------------------------------;
                                ;31133          VA_R[R3]_M[MDR],                 ;Nothing pending or timer service, save
U 129A, 0C85,2592,40B4,C4A7,0000,1 ;31134       RETURN [T]                       ;dest. address & RETURN
                                ;31135
                                ;31136          ;1------------------------------;
                                ;31137          INTPEND OR TIMER?,               ;GPR3_dest. address incase we pack up
                                ;31138          PUSH,NEXT/IE.SERV.IP.TS2,
U 129B, 0085,2592,4AF4,C047,04F7,0 ;31139       R[R3]_M[MDR]
```

```
                    ;31140 .TOC "  Decimal String                    : DS.CVTXX.PACK"
                    ;31141
                    ;31142 ;*****************************************************************************
                    ;31143 ;       Entry point     DS.CVTXX.PACK
                    ;31144 ;
                    ;31145 ;       Input           TEMP0           PC-PCBACK
                    ;31146 ;                       TEMP3           Table address if any
                    ;31147 ;                       TEMP4           Zero extend destination length
                    ;31148 ;                       R0              Zero extended source length
                    ;31149 ;                       (R1 & R3        Source and destination address)
                    ;31150 ;
                    ;31151 ;       Output          R0              Table address if any
                    ;31152 ;                       R1              Source address
                    ;31153 ;                       R2<5-0>         Flags
                    ;31154 ;                         <15-8>        Source length
                    ;31155 ;                         <23-16>       Destination length
                    ;31156 ;                         <31-24>       PC - PCBACK
                    ;31157 ;                       R3              Destination address
                    ;31158 ;
                    ;31159 ;*****************************************************************************
                    ;31160
                    ;31161 DS.CVTXX.PACK:
                    ;31162         ;--------------------------------;
U 1370, 0484,0592,4034,8047,007F,8  ;31163         R[R2]_M[TEMP0]                  ;deltaPC
                    ;31164
U 07F8, 0884,02B7,0004,8047,007F,9  ;31165 7F8:    ;*********FORCE ADDRESS*********;
                    ;31166         R[R2]_RB.RR.8                   ; SHIFT TO MAKE ROOM FOR FLAGS
                    ;31167
U 07F9, 0882,0036,4004,8387,0137,1  ;31168 7F9:    ;*********FORCE ADDRESS*********;
                    ;31169         R[R2].SIZ_FLAGS,SIZE[BYTE]      ; SAVE FLAGS
                    ;31170
                    ;31171         ;--------------------------------;
U 1371, 0886,05BE,4034,0047,0137,2  ;31172         M[TEMP0]_R[R0]                  ;Mask in Source length
                    ;31173
                    ;31174         ;--------------------------------;
U 1372, 0484,03BE,4024,8047,0137,3  ;31175         R[R2]_RB.OR.(M[TEMP0].RL.8)
                    ;31176
                    ;31177         ;--------------------------------;
U 1373, 0884,43BE,4014,8047,0137,4  ;31178         R[R2]_RB.OR.(M[TEMP4].RL.16)    ;Mask in Destination Length
                    ;31179
                    ;31180         ;--------------------------------;
                    ;31181         R[R0]_M[TEMP3],                 ;Load Table address, pack is complete
U 1374, 0084,3592,4034,0047,00FE,2  ;31182         NEXT/TE.PACK.DONE
```

```
                                ;31183 .TOC "  Decimal String                    : DS.CVTXX.UNPACK"
                                ;31184
                                ;31185 ;*****************************************************************************
                                ;31186 ;       Entry point     DS.CVTXX.UNPACK
                                ;31187 ;
                                ;31188 ;       Input           R0              Table address if any
                                ;31189 ;                       R1              Source address
                                ;31190 ;                       R2<7-0>         PC - PCBACK
                                ;31191 ;                         <15-8>        Source length
                                ;31192 ;                         <23-16>       Destination length
                                ;31193 ;                       R3              Destination address
                                ;31194 ;
                                ;31195 ;       Output          TEMP1           Source length
                                ;31196 ;                       TEMP2           Source address
                                ;31197 ;                       TEMP3           Table address if any
                                ;31198 ;                       TEMP4           Destination length
                                ;31199 ;                       TEMP5           PC - PCBACK
                                ;31200 ;                       PC              Address of next opcode
                                ;31201 ;                       MDR             Destination address
                                ;31202 ;
                                ;31203 ;       Subroutines     DS.CVTPS.RESTART
                                ;31204 ;                       DS.CVTSP.RESTART
                                ;31205 ;                       DS.CVTPT.RESTART
                                ;31206 ;                       DS.CVTTP.RESTART
                                ;31207 ;
                                ;31208 ;*****************************************************************************
                                ;31209
                                ;31210 .REGION/IRD1.R1L,IRD1.R1H
                                ;31211 378:    ;*********FORCE ADDRESS*********;
                                ;31212 DS.CVTXX.UNPACK:
                                ;31213         ;000--------------------------------;
U 0378, 0086,12B7,0004,8047,0137,5  ;31214         M[TEMP1]_R[R2].RR.8             ;Get Source length
                                ;31215
                                ;31216 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;31217         ;--------------------------------;
U 1375, 0486,55BE,4034,8047,00FF,3  ;31218         M[TEMP5]_R[R2]                  ; NOP LEFT TO AVOID ROM CHANGE
                                ;31219
                                ;31220 0FF3:   ;*********FORCE ADDRESS*********;
U 0FF3, 0886,52B7,0024,8047,007F,A  ;31221         M[TEMP5]_R[R2].RR.24            ;Get delta PC
                                ;31222
                                ;31223 7FA:    ;*********FORCE ADDRESS*********;
U 07FA, 0C80,05BE,4034,8307,0137,6  ;31224         FLAGS_R[R2]                     ; RESTORE FLAGS
                                ;31225
                                ;31226         ;--------------------------------;
U 1376, 0986,1C12,0037,F847,0137,7  ;31227         M[TEMP1]_MB.AND.ZLIT0[0FF]      ;ZEXT length & delta PC
                                ;31228
                                ;31229         ;--------------------------------;
U 1377, 0D86,5C12,0037,F847,0137,8  ;31230         M[TEMP5]_MB.AND.ZLIT0[0FF]
                                ;31231
                                ;31232         ;--------------------------------;
U 1378, 0086,25BE,4034,4047,0137,9  ;31233         M[TEMP2]_R[R1]                  ; RESTORE SOURCE ADDRESS
```

```
                                ;31234          ;--------------------------------;
U 1379, 0086,35BE,4034,0047,0137,A ;31235       M[TEMP3]_R[R0]                  ; RESTORE TABLE ADDRESS OR ...
                                ;31236
                                ;31237          ;--------------------------------;
U 137A, 0C86,42B7,0014,8047,0137,B ;31238       M[TEMP4]_R[R2].RR.16            ;Restore destination length
                                ;31239
                                ;31240          ;--------------------------------;
U 137B, 0186,4C12,0037,F847,0137,C ;31241       M[TEMP4]_MB.AND.ZLITO[OFF]
                                ;31242
                                ;31243          ;--------------------------------;
                                ;31244          PC_M[PCBACK]+R[TEMP5],          ;Restore PC with deltaPC
U 137C, 0C81,9001,0671,4487,0110,0 ;31245       IR<2-0>?                        ;in TEMP5, which inst. are we?
                                ;31246
                                ;31247 =000     ;000-----------------------------;
                                ;31248          MDR_R[R3]-CONX(1),              ;
U 1100, 0880,073C,0004,C467,002A,6 ;31249       NEXT/DS.CVTPS.RESTART           ;
                                ;31250
                                ;31251          ;001-----------------------------;
                                ;31252          MDR_R[R3],                      ;
U 1101, 0080,05BE,4034,C467,011C,C ;31253       NEXT/DS.CVTSP.RESTART           ;
                                ;31254
                                ;31255 =100     ;100-----------------------------;
                                ;31256          MDR_R[R3],                      ;
U 1104, 0080,05BE,4034,C467,0036,3 ;31257       NEXT/DS.CVTPT.RESTART           ;
                                ;31258
                                ;31259 =110     ;110-----------------------------;
                                ;31260          MDR_R[R3],                      ;
U 1106, 0080,05BE,4034,C467,0037,2 ;31261       NEXT/DS.CVTTP.RESTART           ;
                                ;31262 =
```

```
                                ;31263 .TOC "  Decimal String                    : DS.CVTXX.SETCC"
                                ;31264
                                ;31265 ;*****************************************************************************
                                ;31266 ;       Entry points    DS.CVTXX.SETCC
                                ;31267 ;                       DS.CVTXX.SETCC.06
                                ;31268 ;                       DS.CVTXX.SETCC.07
                                ;31269 ;
                                ;31270 ;       Input           TEMP8           Address of next opcode
                                ;31271 ;                       FLAGS           Intermediate value of condition codes
                                ;31272 ;
                                ;31273 ;       Resources       TEMP5           Temporary for moving flags to PSL<NZVC>
                                ;31274 ;
                                ;31275 ;       Output          PC              Address of next opcode
                                ;31276 ;                       PSL<FPD>        Clear
                                ;31277 ;
                                ;31278 ;       Subroutines     DS.CCPCIRD1.CMPP
                                ;31279 ;
                                ;31280 ;*****************************************************************************
                                ;31281
                                ;31282 =00
                                ;31283 DS.CVTXX.SETCC:
                                ;31284          ;00-----------------------------;
                                ;31285          PC_R[TEMP8],CLEAR FLAG?,       ;
U 11E8, 0010,05BE,4032,0487,0109,C ;31286       NEXT/DS.CVTXX.SETCC.10         ;
                                ;31287
                                ;31288 DS.CVTXX.SETCC.06:
                                ;31289          ;01-----------------------------;
                                ;31290          PC_R[TEMP8],                   ;
U 11E9, 0080,05BE,4032,0487,0109,C ;31291       NEXT/DS.CVTXX.SETCC.10         ;
                                ;31292
                                ;31293          ;10-----------------------------;
                                ;31294          SET FLAG3,                     ;
U 11EA, 0458,0036,4030,0047,011E,8 ;31295       NEXT/DS.CVTXX.SETCC            ;
                                ;31296
                                ;31297 DS.CVTXX.SETCC.07:
                                ;31298          ;11-----------------------------;
                                ;31299          PC_R[TEMP8],                   ;
                                ;31300          CLEAR FLAG3,                   ;
U 11EB, 0018,05BE,44B2,0487,0109,8 ;31301       FLAG2?                         ; IF RESULT NOT ZERO SET PSL<N>
                                ;31302 =0**
                                ;31303 DS.CVTXX.SETCC.08:
                                ;31304          ;0*-----------------------------;
U 1098, 0058,05BE,4032,0487,0109,C ;31305       PC_R[TEMP8],SET FLAG3          ;
                                ;31306
                                ;31307 DS.CVTXX.SETCC.10:
                                ;31308          ;1*-----------------------------;
U 109C, 0CE6,5036,4030,0387,0137,D ;31309       M[TEMP5]_FLAGS,CLEAR FPD       ;
                                ;31310
                                ;31311          ;-------------------------------;
                                ;31312          CC_M[TEMP5],                   ; GPR0_0 and IRD1 in DS.CVTPL  code
U 10?D, 0880,5592,4030,00A7,0134,9 ;31313       NEXT/DS.CVTPL.IRD1             ; Modified for rev94 to save a word
```

```
                        ;31314 .TOC "  Decimal String                    : Misc Subroutines"
                        ;31315
                        ;31316 ;*****************************************************************************
                        ;31317 ;       Entry Points    DS.TABLE.ADDR
                        ;31318 ;
                        ;31319 ;       Input           VA              Destination address
                        ;31320 ;                       TEMP3           Table address
                        ;31321 ;                       TEMP6           Byte to be converted (table offset)
                        ;31322 ;
                        ;31323 ;       Outupt          TEMP7           Save destination address (input VA)
                        ;31324 ;                       VA              Address of desired byte in table
                        ;31325 ;
                        ;31326 ;       This is used by DS.CVTTP and DS.CVTPT when looking up the byte
                        ;31327 ;       in the table.
                        ;31328 ;
                        ;31329 ;*****************************************************************************
                        ;31330
                        ;31331 DS.TABLE.ADDR:
                        ;31332         ;-------------------------------;
u 137E, 0085,B592,4031,C047,0137,F      ;31333         R[TEMP7]_M[VA]                  ;
                        ;31334
                        ;31335         ;-------------------------------;
                        ;31336         VA_M[TEMP3]+R[TEMP6],           ;
u 137F, 0880,3001,00B1,84A7,0000,2      ;31337         RETURN [+2]                     ;
                        ;31338
                        ;31339 ;*****************************************************************************
                        ;31340 ;       Entry Points    DS.INC.VA.BY.2
                        ;31341 ;
                        ;31342 ;       Increment the VA by 2.
                        ;31343 ;
                        ;31344 ;*****************************************************************************
                        ;31345
                        ;31346 DS.INC.VA.BY.2:
                        ;31347         ;-------------------------------;
                        ;31348         VA_M[VA]+2LITO[2],              ;
 ;380, 0181,BC11,00B0,14A7,03FF,E      ;31349         RETURN [-2]                     ;
```

```
;31350 .TOC "  Decimal String                    : ASHP"
;31351
;31352          ;*****************************************************************************
;31353          ;       ASHP cnt.rb, srclen.rw, srcaddr.ab, round.rb,
;31354          ;            dstlen.rw, dstaddr.ab
;31355          ;
;31356          ;       If the count is odd, we shift by 10. to get it even.  Once
;31357          ;       even we shift by bytes (100.) until count is zero.  A negative
;31358          ;       count will cause the round operand to be added after shifting.
;31359          ;
;31360          ;       Input           MDR             cnt.rb
;31361          ;
;31362          ;       Resources       GPR1            Base address, source
;31363          ;                       GPR3            Destination address
;31364          ;                       TEMP3-TEMP6     Packed Decimal data, memory
;31365          ;                                       format, LSB in byte#3 of TEMP3,
;31366          ;                                       MSB in byte#0 of TEMP6
;31367          ;                       TEMP7           Used for Zero (0) fill
;31368          ;                       MTEMP8          Source Length
;31369          ;                       MTEMP9          Destination Length
;31370          ;                       MTEMP10         Absolute value of Count operand
;31371          ;                       FPDOFFSET       ZLITO[15.]
;31372          ;                       RTEMP8          Count operand
;31373          ;                       RTEMP9          Round operand
;31374          ;                       RNUM            Points to TEMPs during I/O
;31375          ;                       DREG            Copy of RNUM, general data
;31376          ;                       QREG            Temp. for DS.WRITE
;31377          ;                       ALUS            Stores sign of source and loop
;31378          ;                                       control.  As count is dec., the
;31379          ;                                       ALUS_polarity of result
;31380          ;                       STEPC           BYTE counter for I/O routines
;31381          ;                       PLATCH          [8.], used to rotate bytes
;31382          ;                       MDR             READs
;31383          ;                       WDR             WRITEs
;31384          ;                       VA              I/O
;31385          ;                       PSL<C>          =0 if even destination
;31386          ;                                       =1 if odd destination
;31387          ;                       FLAG0           DS.READ resource
;31388          ;                                       Also masks out PSL<C>
;31389          ;                       FLAG1           <V> image, initially 0
;31390          ;                       FLAG2           =0 if Count operand is >=0
;31391          ;                                       =1 if Count operand is <0.
;31392          ;                                       Becomes <Z>image when calling
;31393          ;                                       DS.WRITE.
;31394          ;                       FLAG3           =0 if source is pos.,
;31395          ;                                       =1 if source is neg.
;31396          ;                                       (<N> image)
;31397          ;
;31398          ;       Output          GPR0            Zero (0)
;31399          ;                       GPR1            Base address, source
;31400          ;                       GPR2            Zero (0)
;31401          ;                       GPR3            Base address, destination
;31402          ;                       PSL<NZVC>       modified
;31403          ;*****************************************************************************
```

```
                                ;31404 .TOC     "       Decimal String                  :       Initialization"
                                ;31405
                                ;31406 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;31407 =0
                                ;31408 DS.ASHP:
                                ;31409          ;0-------------------------------;
                                ;31410          PUSH,SIZE[BYTE],                ;Do PUSH to return after last operand,
                                ;31411          R[TEMP8]_SEXT(M[MDR]),          ;Store COUNT operand, read srclen.rw
u 0326, 0085,2E5E,0182,0047,0410,0  ;31412      LOD INC BRA?,NEXT/OS.RED        ;(We will continue at DS.ASHP.2:)
                                ;31413
                                ;31414          ;1-------------------------------;
                                ;31415          D_M[TEMP1].OR.R[TEMP3],         ;Here after last operand read, OR
u 0327, 0880,1002,6030,C047,0138,1  ;31416      NEXT/DS.ASHP.LENGTH             ;Source & Destination lengths
                                ;31417
                                ;31418 =00
                                ;31419 DS.ASHP.2:
                                ;31420          ;00------------------------------;
                                ;31421          R[TEMP1]_SEXT(M[MDR]),          ;TEMP1_srclen.rw, read source address
                                ;31422          LOD INC BRA?,NEXT/OS.ADD,
u 02B4, 0C85,2E5E,0190,4047,0412,0  ;31423      PUSH,SIZE[WORD]
                                ;31424
                                ;31425          ;01------------------------------;
                                ;31426          R[TEMP2]_M[MDR],PUSH,           ;TEMP2_srcaddr.ab, read round.rb
u 02B5, 0085,2592,41B0,8047,0410,0  ;31427      LOD INC BRA?,NEXT/OS.RED
                                ;31428
                                ;31429          ;10------------------------------;
                                ;31430          R[TEMP9]_ZEXT(M[MDR]),          ;RTEMP9_ROUND operand, read dstlen.rw
                                ;31431          SIZE[BYTE],PUSH,LOD INC BRA?,
u 02B6, 0485,259E,4182,4047,0410,0  ;31432      NEXT/OS.RED
                                ;31433
                                ;31434          ;11------------------------------;
                                ;31435          R[TEMP3]_SEXT(M[MDR]),          ;TEMP3_destination length, anything
u 02B7, 0485,2E5E,0B10,D8E7,0129,C  ;31436      SIZE[WORD],IP.TS?               ;pending?
                                ;31437
                                ;31438 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;31439 =0
                                ;31440          ;0-------------------------------;
                                ;31441          M[TEMP7]_CC_ZLIT0[0],           ;No, or timer service, shift-in temp.
u 129C, 0186,7C37,01B0,00A7,0012,0  ;31442      LOD INC BRA?,NEXT/OS.ADD        ;& CC(guess even dest.) zero, read
                                ;31443                                          ;dstaddr.ab without PUSH, return to
                                ;31444                                          ;DS.ASHP:+1
                                ;31445
                                ;31446          ;1-------------------------------;
                                ;31447          INTPEND OR TIMER?,PUSH,         ;Yes, see what it is, roll back if int.
u 129D, 0480,0036,4AF0,0047,04F7,0  ;31448      NEXT/IE.SERV.IP.TS2             ;pending
```

```
                                ;31449          ;*************************************************************************
                                ;31450          ; Operand reads are done, DREG has OR of lengths.
                                ;31451          ;*************************************************************************
                                ;31452
                                ;31453 DS.ASHP.LENGTH:
                                ;31454          ;-------------------------------;
                                ;31455          WB_D.ANDNOT.ZLIT0[31.],         ;Are lengths legal?
U 1381, 0980,0C33,8A70,F847,0129,E ;31456       WX.NE.0?
                                ;31457
                                ;31458 =0
                                ;31459          ;0-------------------------------;
                                ;31460          M[TEMP8]_R[TEMP1],IP.TS?,       ;Yes, store source length,
U 129E, 0086,85BE,4B30,58E7,012A,0 ;31461       NEXT/DS.ASHP.INTC               ;anything pending?
                                ;31462
                                ;31463          ;1-------------------------------;
U 129F, 0C80,0036,4030,0047,00FF,8 ;31464       NEXT/IE.OPER.FAULT              ;No, **reserved operand fault**
                                ;31465
                                ;31466 =0
                                ;31467 DS.ASHP.INTC:
                                ;31468          ;0-------------------------------;
                                ;31469          R[R1]_M[TEMP2],SET FPD,         ;Nothing pending or timer service,
U 12A0, 00EC,2592,4034,4047,0138,2 ;31470       NEXT/DS.ASHP.INIT               ;load source address & FPD_1
                                ;31471
                                ;31472          ;1-------------------------------;
                                ;31473          PUSH,INTPEND OR TIMER?,         ;Yes, return -1 or roll back
U 12A1, 0480,0036,4AF0,0047,04F7,0 ;31474       NEXT/IE.SERV.IP.TS2
                                ;31475
                                ;31476
                                ;31477 DS.ASHP.INIT:
                                ;31478          ;-------------------------------;
U 1382, 0085,2592,4034,C047,0138,3 ;31479       R[R3]_M[MDR]                    ;GPR3_Destination address
                                ;31480
                                ;31481          ;-------------------------------;
U 1383, 0C86,95BE,42B0,C047,012A,2 ;31482       M[TEMP9]_R[TEMP3],WB<0>?        ;MTEMP9_destination length, is it odd?
                                ;31483
                                ;31484
                                ;31485          ;*************************************************************************
                                ;31486          ; After FPD unpacking we reenter at DS.ASHP.REE: directly to split on
                                ;31487          ; polarity of Count.  Initialization splits on evenness/oddness of dest.
                                ;31488          ;*************************************************************************
                                ;31489 =0
                                ;31490 DS.ASHP.REE:
                                ;31491 DS.ASHP.CNT:
                                ;31492          ;0-------------------------------;
                                ;31493          M[TEMP10]_R[TEMP8],             ;(no, even,) What is sign of count?
                                ;31494          SIGND CMP?,NEXT/DS.ASHP.CSIGN,
U 12A2, 0C16,A5BE,4B62,0047,091E,D 351* ;31495  SIZE[LONG],CLEAR FLAG2
                                ;31496
                                ;31497          ;1-------------------------------;
U 12A3, 0980,0C37,0030,08A7,012A,2 ;31498       CC_ZLIT0[1],NEXT/DS.ASHP.CNT    ;Yes, PSL<C>_1, check pointer
```

```
                                ;31499 .TOC     "       Decimal String                  :       Source READ and sign decode"
                                ;31500
                                ;31501 =01
                                ;31502 DS.ASHP.CSIGN:
                                ;31503          ;01-------------------------------;
                                ;31504          M[FPDOFFSET]_ZLIT0[15.],        ;(Count>=0), load IANDE ASHP offset
U 11ED, 0986,CC37,0030,7847,0138,4  ;31505      NEXT/DS.ASHP.PRE.READ
                                ;31506
                                ;31507          ;11-------------------------------;
                                ;31508          SET FLAG2,M[TEMP10]_-MB,        ;Count<0, set FLAG, make it pos.
U 11EF, 0856,A003,003D,8047,011E,D  ;31509      NEXT/DS.ASHP.CSIGN
                                ;31510
                                ;31511
                                ;31512 DS.ASHP.PRE.READ:
                                ;31513          ;-------------------------------;
U 1384, 0D80,0EF6,4030,4047,0138,5  ;31514      PL_[8.]                         ;Load PLATCH for byte rotates
                                ;31515
                                ;31516          ;-------------------------------;
                                ;31517          STEPC_D_M[TEMP8].SR.1,          ;L/2, source, STEPC is loaded
U 1385, 0080,8001,A03D,8107,012A,5  ;31518      NEXT/DS.ASHP.SOURCE.IN
                                ;31519
                                ;31520 =0
                                ;31521          ;0--------------------------------;
                                ;31522          STEPC_Q_M[TEMP9].SR.1,ALUS?,    ;STEPC_Q_L/2 for future DS.WRITE call,
U 12A4, 0880,9E52,9B70,1907,011F,1  ;31523      NEXT/DS.ASHP.LOOP.SGN           ;what is sign of source?
                                ;31524
                                ;31525 DS.ASHP.SOURCE.IN:
                                ;31526          ;1--------------------------------;
U 12A5, 0C80,00A1,0034,44A7,0505,D  ;31527      PUSH,VA_D+R[R1]+1,NEXT/DS.READ  ;Point VA past sign byte, read source
                                ;31528
                                ;31529
                                ;31530 =01
                                ;31531 DS.ASHP.LOOP.SGN:
                                ;31532          ;01-------------------------------;
                                ;31533          M[TEMP10]_MB-1,SIZE[LONG],      ;Positive source, dec. count, is result
                                ;31534          ALUS_SIGND,CLEAR FLAG3,         ;odd or even?, & ALUS_polarity of result
U 11F1, 001E,A080,02AD,98C7,091F,4 344*  ;31535 WB<0>?,NEXT/DS.ASHP.LOOP.SPL
                                ;31536
                                ;31537          ;11-------------------------------;
                                ;31538          M[TEMP10]_MB-1,SIZE[LONG],      ;Negative source, dec. count, is result
U 11F3, 085E,A080,02AD,98C7,091F,4 344*  ;31539 ALUS_SIGND,WB<0>?,SET FLAG3       ;odd or even, & ALUS_polarity of result
```

```
                                ;31540 .TOC     "       Decimal String                  :       Beginning & Loop of ASHP shi
                                ;31541
                                ;31542          ;*************************************************************************
                                ;31543          ; ASHP shift loop begins here
                                ;31544          ;*************************************************************************
                                ;31545 =00
                                ;31546 DS.ASHP.LOOP.SPL:
                                ;31547          ;00-----------------------------;
                                ;31548          M[TEMP10]_MB-CONX(2),ALUS_SIGND,;Count is 0,2,4..., dec. count for next
U 11F4, 0C86,A710,0490,18C7,0515,B  ;31549      FLAG2?,NEXT/DS.ASHP.NIB,PUSH    ;(byte) decision, split to nib. code,
                                ;31550                                          ;PUSH for DS.xxx10 RETURN [+2]
                                ;31551
                                ;31552          ;01-----------------------------;
                                ;31553          M[TEMP10]_MB-1,                 ;Count is -1,1,3...it was even & is
U 11F5, 0486,A080,0B7D,9847,011F,9  ;31554      ALUS?,NEXT/DS.ASHP.BYT.SPL      ;again, is it neg or pos.?
                                ;31555
                                ;31556          ;10-----------------------------;
U 11F6, 0480,0036,4B70,1847,011F,9  ;31557      ALUS?                           ;We've shifted source by a nibble,
                                ;31558                                          ;what is polarity of this pass?
                                ;31559 =
                                ;31560
                                ;31561          ;*************************************************************************
                                ;31562          ; Here from first Count decrement (original count was even) or after
                                ;31563          ; a byte-shift pass or the nibble shift pass from an odd count.  ALUS
                                ;31564          ; was loaded with the signd compare of the Count decrement (by 2) for
                                ;31565          ; this pass.
                                ;31566          ; =01 is Count is positive or zero, it was positive, decrement it for
                                ;31567          ; next pass & shift by a byte.
                                ;31568          ; =11 is Count was negative after being decremented, we just finished
                                ;31569          ; our last pass.
                                ;31570          ;*************************************************************************
                                ;31571 =01
                                ;31572 DS.ASHP.BYT.SPL:
                                ;31573          ;01-----------------------------;
                                ;31574          M[TEMP10]_MB-CONX(2),ALUS_SIGND,;Dec. count & store polarity for next
U 11F9, 0C86,A710,0490,18C7,010E,A  ;31575      FLAG2?,NEXT/DS.ASHP.BYT.SHF     ;pass through, which way are we ASHing?
                                ;31576
                                ;31577          ;11-----------------------------;
                                ;31578          M[TEMP7]_R[TEMP9].RR.8,FLAG2?,  ;Rotate round operand to proper byte,
U 11FB, 0886,72B7,0482,4047,0116,8  ;31579      NEXT/DS.ASHP.DONE               ;which way are we shifting?
                                ;31580 =
```

```
                                ;31581          ;*************************************************************************
                                ;31582          ; Byte shifting.  =0** is Count operand is >0, =1** is Count operand <0.
                                ;31583          ;*************************************************************************
                                ;31584 =0**
                                ;31585 DS.ASHP.BYT.SHF:
                                ;31586          ;0**-------------------------------;
                                ;31587          WB_M[TEMP6].AND.ZLIT0[OFF],     ;Multiplying, will we shift anything
U 10EA, 0980,6C12,0A77,F847,012A,8  ;31588      WX.NE.0?,NEXT/DS.ASHP.BYT.BIG   ;out?
                                ;31589
                                ;31590          ;1**-------------------------------;
U 10EE, 0486,38F7,0031,0047,0138,6  ;31591      M[TEMP3]_(R[TEMP4] MB).RL.P     ;Dividing, shift 4 TEMPs by a byte
                                ;31592                                          ;toward low end
                                ;31593
                                ;31594          ;-------------------------------;
U 1386, 0C86,48F7,0031,4047,0138,7  ;31595      M[TEMP4]_(R[TEMP5] MB).RL.P
                                ;31596
                                ;31597          ;-------------------------------;
                                ;31598          M[TEMP5]_(R[TEMP6] MB).RL.P,    ;Int. check here
U 1387, 0486,58F7,0B31,98E7,012A,6  ;31599      IP.TS?
                                ;31600
                                ;31601 =0
                                ;31602          ;0-------------------------------;
                                ;31603          M[TEMP6]_(R[ZERO] MB).RL.P,     ;Nothing or timer service, loop back
U 12A6, 0086,68F7,0B7D,9847,011F,9  ;31604      ALUS?,NEXT/DS.ASHP.BYT.SPL
                                ;31605
                                ;31606          ;1-------------------------------;
                                ;31607          INTPEND OR TIMER?.PUSH,         ;RETURN-1 or branch to IE.PACK.DONE
U 12A7, 0480,0036,4AF0,0047,04F7,0  ;31608      NEXT/IE.SERV.IP.TS2
```

```
                                ;31609          ;*********************************************************************
                                ;31610          ; Byte shift, multiplying.  =0 is nothing will be shifted out from
                                ;31611          ; TEMP6, =1 is something will be lost, we have overflow.  We shift
                                ;31612          ; 4 TEMPs by a byte from high end.
                                ;31613          ;*********************************************************************
                                ;31614 =0
                                ;31615 DS.ASHP.BYT.BIG:
                                ;31616          ;0-----------------------------;
                                ;31617          R[TEMP6]_(M[TEMP5] RB).RR.P,
U 12A8, 0C84,5137,0031,8047,0138,8  ;31618      NEXT/DS.ASHP.BYT.BIG.2
                                ;31619
                                ;31620          ;1-----------------------------;
                                ;31621          SET FLAG1,                      ;<V> image_1
U 12A9, 044C,5137,0031,8047,0138,8  ;31622      R[TEMP6]_(M[TEMP5] RB).RR.P
                                ;31623
                                ;31624 DS.ASHP.BYT.BIG.2:
                                ;31625          ;-----------------------------;
U 1388, 0084,4137,0031,4047,0138,9  ;31626      R[TEMP5]_(M[TEMP4] RB).RR.P
                                ;31627
                                ;31628          ;-----------------------------;
                                ;31629          R[TEMP4]_(M[TEMP3] RB).RR.P,    ;Anything pending?
U 1389, 0484,3137,0B31,18E7,012A,A  ;31630      IP.TS?
                                ;31631
                                ;31632 =0
                                ;31633          ;0-----------------------------;
                                ;31634          R[TEMP3]_(M[TEMP7] RB).RR.P,    ;No, or timer service, TEMP7 is zero,
U 12AA, 0084,7137,0B70,D847,011F,9  ;31635      ALUS?,NEXT/DS.ASHP.BYT.SPL      ;loop back
                                ;31636
                                ;31637          ;1-----------------------------;
                                ;31638          INTPEND OR TIMER?,PUSH,         ;Yes, RETURN-1 or branch to IE.PACK.DONE
U 12AB, 0480,0036,4AF0,0047,04F7,0  ;31639      NEXT/IE.SERV.IP.TS2
```

```
                                ;31640          ;*************************************************************************
                                ;31641          ; Nibble (10.) shift being done.  Here on FLAG2? BUT, =0** is count>0,
                                ;31642          ; we are multiplying.  =1** is we are dividing.  PUSH for multiply
                                ;31643          ; and divide routines was made at DS.ASHP.LOOP.SPL: and we will RETURN
                                ;31644          ; +2.
                                ;31645          ;*************************************************************************
                                ;31646 =0**
                                ;31647 DS.ASHP.NIB:
                                ;31648          ;0**-----------------------------;
                                ;31649          WB_M[TEMP6].AND.ZLIT0[0F0],     ;Will we lose anything on shift-out?
U 115B, 0D80,6C12,0A77,8047,012A,C ;31650       WX.NE.0?,NEXT/DS.ASHP.NIB.BIG
                                ;31651
                                ;31652          ;1**-----------------------------;
U 115F, 0480,0036,4030,0047,012B,8 ;31653       NEXT/DS.DIV10                   ;Divide-by-10.
                                ;31654
                                ;31655
                                ;31656 =0
                                ;31657 DS.ASHP.NIB.BIG:
                                ;31658          ;0-------------------------------;
U 12AC, 0C80,0036,4030,0047,012B,C ;31659       NEXT/DS.MUL10                   ;Nothing lost, do multiply
                                ;31660
                                ;31661          ;1-------------------------------;
U 12AD, 0448,0036,4030,0047,012B,C ;31662       SET FLAG1,NEXT/DS.MUL10         ;Overflow, ditto
```

```
                                ;31663 .TOC     "       Decimal String                  :       Adding Round, writing dest."
                                ;31664
                                ;31665          ;*************************************************************************
                                ;31666          ; Shifting is done, Round operand is rotated to correct byte for add
                                ;31667          ; if needed, STEPC & QREG are loaded for DS.WRITE call.  Here on
                                ;31668          ; FLAG2? split, =0** is positive shift, =1** is negative shift.
                                ;31669          ;*************************************************************************
                                ;31670 =0**
                                ;31671 DS.ASHP.DONE:
                                ;31672          ;0**-----------------------------;
                                ;31673          WB_0-CONX(1),                   ;Force ALUS<1>_1, what was sign of
                                ;31674          ALUS_UNSGN,FLAG3?,              ;source?
U 1168, 0080,0734,04C0,10C7,012A,E ;31675       NEXT7DS.ASHP.SIGN
                                ;31676
                                ;31677          ;1**-----------------------------;
U 116C, 0086,3001,4021,C047,0138,A ;31678       M[TEMP3]_(MB+R[TEMP7]).BCD      ;Add round to most signif.
                                ;31679                                          ;digit shifted out
                                ;31680
                                ;31681          ;-------------------------------;
U 138A, 0C86,4041,402D,8047,0138,B ;31682       M[TEMP4]_(MB+ALKC).BCD          ;And add ALK<C> up the string
                                ;31683
                                ;31684          ;-------------------------------;
U 138B, 0086,5041,402D,8047,0138,C ;31685       M[TEMP5]_(MB+ALKC).BCD
                                ;31686
                                ;31687          ;-------------------------------;
U 138C, 0986,3C93,8030,7847,0138,D ;31688       M[TEMP3]_MB.ANDNOT.ZLIT24[0F]   ;Zero sign nibble position
                                ;31689
                                ;31690          ;-------------------------------;
                                ;31691          M[TEMP6]_(MB+ALKC).BCD,         ;ALUS_CARRY data, what was source
U 138D, 0086,6041,44ED,90C7,012A,E ;31692       ALUS_UNSGN,FLAG3?               ;sign?
                                ;31693
                                ;31694
                                ;31695          ;*************************************************************************
                                ;31696          ; FLAG0=0 (count was positive) split does FLAG3? BUT to mask sign.  If
                                ;31697          ; ALUS<1>=0, we have ov from adding round operand, if ALUS<1>=1, no
                                ;31698          ; ov (Count>0 forced ALUS<1>_1).
                                ;31699          ;*************************************************************************
                                ;31700 =0
                                ;31701 DS.ASHP.SIGN:
                                ;31702          ;0-------------------------------;
                                ;31703          M[TEMP3]_MB.OR.ZLIT24[0C],      ;Source is positive
U 12AE, 0586,3C92,4030,6047,012B,1 ;31704       NEXT/DS.ASHP.SIGN.2
                                ;31705
                                ;31706          ;1-------------------------------;
                                ;31707          M[TEMP3]_MB.OR.ZLIT24[0D],      ;Source is negative
U 12AF, 0186,3C92,4030,6847,012B,1 ;31708       NEXT/DS.ASHP.SIGN.2
```

```
                                ;31709          ;*************************************************************************
                                ;31710          ; PUSH is made at DS.ASHP.SIGN.2 for DS.WRITE call(s)
                                ;31711          ;*************************************************************************
                                ;31712 =0
                                ;31713          ;0-----------------------------;
                                ;31714          D_M[TEMP9].SR.1,FLAG<2-0>?,     ;RETURN from WRITE, D_L/2, split on
U 12B0, 0880,9591,A5B0,0047,0110,8 ;31715       NEXT/DS.ASHP.FLAGS              ;<ZV0> to =000, =010, =100, or =110
                                ;31716
                                ;31717 DS.ASHP.SIGN.2:
                                ;31718          ;1-----------------------------;
U 12B1, 0850,05BE,4B74,DCA7,051F,D ;31719       PUSH,ALUS?,VA_R[R3],SET FLAG2   ;Load base dest. address for DS.WRITE,
                                ;31720                                          ;Init. <Z>image
                                ;31721
                                ;31722
                                ;31723 =01
                                ;31724          ;01----------------------------;
U 11FD, 0448,0036,4030,0047,012C,E ;31725       SET FLAG1,NEXT/DS.WRITE         ;CARRY=1, <V>image_1, write data
                                ;31726
                                ;31727          ;11----------------------------;
U 11FF, 0C80,0036,4030,0047,012C,E ;31728       NEXT/DS.WRITE                   ;Ok, write data
```

```
                                ;31729 .TOC     "       Decimal String                  :       Setting CC"
                                ;31730
                                ;31731          ;*************************************************************************
                                ;31732          ; BUT table from FLAG<2-0>? split above and a FLAG3? split
                                ;31733          ; from =100 (overlapping).  FLAG0=0 for FLAG<2-0>? split.
                                ;31734          ;
                                ;31735          ; DIVP branches directly to DS.ASHP.FLAGS to set CC & finish inst.
                                ;31736          ;*************************************************************************
                                ;31737 =000
                                ;31738 DS.ASHP.FLAGS:
                                ;31739          ;000-------------------------------;
U 1108, 0884,0036,4034,8387,0138,E  ;31740          R[R2]_FLAGS,NEXT/DS.CCPCIRD1     ;Dest. is nonzero, nonov, ok
                                ;31741                                           ;(Also from FLAG3? split)
                                ;31742
                                ;31743          ;001-------------------------------;
                                ;31744          WRITE ZLIT0[0C],SIZE[BYTE],      ;(From FLAG3? split) WRITE +0 to dest.,
U 1109, 0118,0C37,0000,65D8,0110,8  ;31745          CLEAR FLAG3,NEXT/DS.ASHP.FLAGS   ;<N>image_0, finish inst.
                                ;31746
                                ;31747          ;010-------------------------------;
U 110A, 0884,0036,4034,8387,0138,E  ;31748          R[R2]_FLAGS,NEXT/DS.CCPCIRD1     ;Dest. is nonzero, ov, ok
                                ;31749
                                ;31750 =100
                                ;31751          ;100-------------------------------;
                                ;31752          VA_D+R[R3],FLAG3?,               ;Dest. is zero, no ov, is it negative?
U 110C, 0880,0021,04F4,C4A7,0110,8  ;31753          NEXT/DS.ASHP.FLAGS               ;(BUT to =000 or =001)
                                ;31754
                                ;31755 =110
                                ;31756          ;110-------------------------------;
U 110E, 0418,0036,4030,0047,0110,8  ;31757          CLEAR FLAG3,NEXT/DS.ASHP.FLAGS   ;Dest. is zero, ov, dest. is ok, CC<N>_0
                                ;31758 =
```

```
                        ;31759          ;*************************************************************************
                        ;31760          ; GPR2 has CC image, dest. is finished.  This is a common instruction
                        ;31761          ; exit for all Packed Decimal instructions.  Note that certain
                        ;31762          ; instructions listed here enter ASHP code before DS.CCPCIRD1.
                        ;31763          ;
                        ;31764          ; ADDP<46> and SUBP<46> enter directly at DS.CCPCIRD1.
                        ;31765          ; ASHP enters directly at DS.CCPCIRD1.
                        ;31766          ; CVTxx instructions finishing through DS.CTVXX.SETCC code enter
                        ;31767          ; at DS.CCPCIRD1.CMPP.
                        ;31768          ; DIVP enters directly at DS.CCPCIRD1.
                        ;31769          ; MOVP enters directly at DS.CCPCIRD1.
                        ;31770          ; MULP enters directly at DS.CCPCIRD1.
                        ;31771          ;*************************************************************************
                        ;31772
                        ;31773 DS.CCPCIRD1:
                        ;31774          ;--------------------------------;
U 138E, 0C80,05BE,4034,80A7,0138,F      ;31775          CC_R[R2]                        ;(MULP,ASHP,DIVP,MOVP,ADDP<46>,SUBP<46>)
                        ;31776                                          ;copy image to CC
                        ;31777
                        ;31778          ;--------------------------------;
                        ;31779          R[R2]_0,CLEAR FPD,              ;Fix GPR2, clear FPD
U 138F, 04E4,05B7,0034,3047,0134,9      ;31780          NEXT/DS.CVTPL.IRD1              ; R[R0]_0 IRD1 is available else where
                        ;31781
                        ;31782          ;++++++++++++++++++
                        ;31783          ; the next word loc 1390 is removed for mm fix
                        ;31784          ;++++++++++++++++++
                        ;31785
                        ;31786 ;DS.CCPCIRD1.CMPP:
                        ;31787 ;        ;--------------------------------;
                        ;31788 ;        R[R0]_0,IRD1                    ;(CVTxx) GPR0 is ok, IRD1
```

```
;31789 .TOC     "       Decimal String                  : DS.DS.PCK routines"
;31790
;31791          ;*****************************************************************************
;31792          ; Pack routines for some Decimal String instructions.
;31793          ; Instruction:  FPDOFFSET:      Enters at:
;31794          ; ADDP4         14.             DS.DS.PCK.GPR2.2-3
;31795          ; SUBP4         14.             DS.DS.PCK.GPR2.2-3
;31796          ; ADDP6         14.             DS.DS.PCK.GPR2.2-3
;31797          ; SUBP6         14.             DS.DS.PCK.GPR2.2-3
;31798          ; ASHP          15.             DS.DS.PCK.GPR2.3CON
;31799          ; MULP          16.             DS.DS.PCK.MULP
;31800          ; MOVP          17.             DS.DS.PCK.GPR0.CON
;31801          ; CMPP3         20.             DS.DS.PCK.CMPP34
;31802          ; CMPP4         20.             DS.DS.PCK.CMPP34
;31803          ; CMPP4         18.             DS.DS.PCK.CMPP4
;31804          ;       (leading byte READs (seperate flows))
;31805          ;
;31806          ; Some of the following inputs are set up in the IANDE FPDOFFSET
;31807          ; BUT table.  M=MOVP, C3=CMPP3, C4=CMPP4 in sep. flows, C6=CMPP4 when
;31808          ; in CMPP3 code, +=<ADDSUB>P<46>, *=MULP, ^=ASHP:
;31809          ;
;31810          ;       Input           GPR0    MTEMP8 (Source 1 L) in byte#1, 0 fill
;31811          ;                               in other bytes (*).
;31812          ;                               PC (MC3C6).
;31813          ;                               deltaPC (Istream) (C4)
;31814          ;                       GPR2    MTEMP9 (Source 2 L) (**^)
;31815          ;                       PC      Points past Istream (**^)
;31816          ;                       PCBACK  Opcode +2 (MC3C4C6**^)
;31817          ;                       TEMP1   READing base address, source #1 (C6).
;31818          ;                               Source 1 Length (C4).
;31819          ;                       TEMP2   READing base address, source #2 (C6)
;31820          ;                       TEMP3   RTEMP8 in byte#1 (*).
;31821          ;                               Source 2 Length (C4).
;31822          ;                       MTEMP8  Source 1 L (MC3C6+^)
;31823          ;                       MTEMP9  Source 2 L (+^)
;31824          ;                       RTEMP8  Dest. L or Count operand (+^)
;31825          ;                       RTEMP9  Round operand (^)
;31826          ;                       FLAGS   FLAGS (MC3C4C6+*^)
;31827          ;
;31828          ;       Resources       TEMP3   Data rotation & calc.
```

```
;31829          ;       Output          For all instructions except CMPP4 during
;31830          ;                       seperate flows:
;31831          ;                       GPR0 #0 FLAGS
;31832          ;                            1 MTEMP8
;31833          ;                            2 <Unused>
;31834          ;                            3 PC-PCBACK
;31835          ;                       GPR2 #0 MTEMP9 (not MOVP, CMPP<34>)
;31836          ;                            1 RTEMP8 (not MOVP, CMPP<34>)
;31837          ;                            2 RTEMP9 (ASHP only)
;31838          ;                       For CMPP4 during seperate (leading byte READs)
;31839          ;                       flows:
;31840          ;                       GPR0 #0 TEMP3 (Length #2)
;31841          ;                            1 TEMP1 (Length #1)
;31842          ;                            2 One (1)
;31843          ;                            3 PC-PCBACK
;31844          ;                       GPR2 #0 "Base address"-Base address (src.#1)
;31845          ;                            1 "Base address"-Base address (src.#2)
;31846          ;
;31847          ; Note that some instructions change FPDOFFSET values during the course
;31848          ; of execution, eg. "MULP" looks & packs like "ADDP6" during
;31849          ; initialization.
;31850          ;*************************************************************************
```

```
                                        ;31851 DS.DS.PCK.GPR0.CON:
                                        ;31852          ;--------------------------------;
U 1391, 0485,9003,0034,0047,0139,2      ;31853          R[R0]_RB-M[PCBACK]              ;Calc. deltaPC
                                        ;31854
                                        ;31855 DS.DS.PCK.GPR0.CON.A:
                                        ;31856          ;--------------------------------;
U 1392, 0C84,02B7,0004,0047,0139,3      ;31857          R[R0]_RB.RR.8
                                        ;31858
                                        ;31859          ;--------------------------------;
U 1393, 0484,83BE,4024,0047,0139,4      ;31860          R[R0]_RB.OR.(M[TEMP8].RR.24)
                                        ;31861
                                        ;31862 DS.DS.PCK.MULP.EX:
                                        ;31863          ;--------------------------------;
                                        ;31864          R[R0].SIZ_FLAGS,SIZE[BYTE],     ;FLAGS, all done
U 1394, 0482,0036,4004,0387,00FE,2      ;31865          NEXT/IE.PACK.DONE
                                        ;31866
                                        ;31867
                                        ;31868
                                        ;31869 DS.DS.PCK.GPR2.2-3:
                                        ;31870          ;--------------------------------;
U 1395, 0C86,32B7,0022,0047,0139,6      ;31871          M[TEMP3]_R[TEMP8].RR.24         ;Move & mask RTEMP8 data (0-1F)
                                        ;31872
                                        ;31873          ;--------------------------------;
                                        ;31874          R[R2]_M[TEMP3].OR.RB,           ;Next loc. is in IANDE BUT table
U 1396, 0484,3002,4034,8047,00FD,1      ;31875          NEXT/DS.DS.PCK.GPR0
                                        ;31876
                                        ;31877
                                        ;31878
                                        ;31879 DS.DS.PCK.GPR2.3CON:
                                        ;31880          ;--------------------------------;
U 1397, 0086,32B7,0012,4047,0139,8      ;31881          M[TEMP3]_R[TEMP9].RR.16         ;RTEMP9 data into position
                                        ;31882
                                        ;31883          ;--------------------------------;
U 1398, 0C84,3002,4034,8047,0139,9      ;31884          R[R2]_M[TEMP3].OR.RB
                                        ;31885
                                        ;31886          ;--------------------------------;
U 1399, 0886,35BE,4032,0047,0139,A      ;31887          M[TEMP3]_R[TEMP8]               ;Count operand can be anything, mask
                                        ;31888                                          ;out top 3 bytes
                                        ;31889
                                        ;31890          ;--------------------------------;
                                        ;31891          R[TEMP8]_ZEXT(M[TEMP3]),
                                        ;31892          SIZE[BYTE],
U 139A, 0884,359E,4002,0047,0139,5      ;31893          NEXT/DS.DS.PCK.GPR2.2-3
```

```
                                ;31894          ;*****************************************************************************
                                ;31895          ; CMPP3-CMPP4 pack during common flows enters here with GPR0 containing
                                ;31896          ; PC on an IR<2-0>? split.  =101 is CMPP3, =111 is CMPP4.
                                ;31897          ;*****************************************************************************
                                ;31898 =101
                                ;31899 DS.DS.PCK.CMPP34:
                                ;31900          ;101-------------------------------;
                                ;31901          R[R0]_RB-M[PCBACK],
U 1035, 0485,9003,0034,0047,0139,2 ;31902       NEXT/DS.DS.PCK.GPR0.CON.A
                                ;31903
                                ;31904          ;111-------------------------------;
U 1037, 0486,1000,0034,4047,0139,B ;31905       M[TEMP1]_MB-R[R1]               ;Calc. "base address" offset, source 1
                                ;31906
                                ;31907          ;---------------------------------;
U 139B, 0886,2000,0034,C047,0139,C ;31908       M[TEMP2]_MB-R[R3]               ;Ditto, source 2
                                ;31909
                                ;31910          ;---------------------------------;
U 139C, 0084,1592,4034,8047,0139,D ;31911       R[R2]_M[TEMP1]                  ;GPR2_Offsets #1 & #2
                                ;31912
                                ;31913          ;---------------------------------;
                                ;31914          R[R2]_RB.OR.(M[TEMP2].RR.24),
U 139D, 0084,23BE,4024,8047,0139,1 ;31915       NEXT/DS.DS.PCK.GPR0.CON
                                ;31916
                                ;31917
                                ;31918 DS.DS.PCK.CMPP4:
                                ;31919          ;---------------------------------;
U 139E, 0C84,02B7,0004,0047,0139,F ;31920       R[R0]_RB.RR.8                   ;(PC-PCBACK) to byte #3
                                ;31921
                                ;31922          ;---------------------------------;
U 139F, 0884,3002,4034,0047,013A,0 ;31923       R[R0]_M[TEMP3].OR.RB            ;Mask in Source 2 length
                                ;31924
                                ;31925          ;---------------------------------;
U 13A0, 0D86,3D37,0030,0847,013A,1 ;31926       M[TEMP3]_ZLIT16[1]              ;Mask in int. code 1
                                ;31927
                                ;31928          ;---------------------------------;
U 13A1, 0084,3002,4034,0047,013A,2 ;31929       R[R0]_M[TEMP3].OR.RB
                                ;31930
                                ;31931          ;---------------------------------;
                                ;31932          R[R0]_RB.OR.(M[TEMP1].RR.24),   ;Source 1 length, packing is complete
U 13A2, 0484,13BE,4024,0047,00FE,2 ;31933       NEXT/IE.PACK.DONE
                                ;31934
                                ;31935
                                ;31936 DS.DS.PCK.MULP:
                                ;31937          ;---------------------------------;
U 13A3, 0484,3002,4034,8047,013A,4 ;31938       R[R2]_M[TEMP3].OR.RB
                                ;31939
                                ;31940          ;---------------------------------;
U 13A4, 0885,A592,4030,C047,013A,5 ;31941       R[TEMP3]_M[PC]                  ;Cal. deltaPC
                                ;31942
                                ;31943          ;---------------------------------;
U 13A5, 0885,9003,0030,C047,013A,6 ;31944       R[TEMP3]_RB-M[PCBACK]
                                ;31945
                                ;31946          ;---------------------------------;
                                ;31947          R[R0]_RB.OR.(M[TEMP3].RR.8),
U 13A6, 0484,33BE,4004,0047,0139,4 ;31948       NEXT/DS.DS.PCK.MULP.EX
```

```
;31949 .TOC     "       Decimal String                  : DS.DS.UNP"
;31950
;31951          ;*************************************************************************
;31952          ; ASHP pack is a DS standard pack with no int. code needed:
;31953          ; inst. is restarted at DS.ASHP.REE:.
;31954          ;
;31955          ; MOVP pack is DS-Standard, GPR0 only, no int. code needed.
;31956          ; Inst. is restarted at DS.MOVP.REE:.
;31957          ;
;31958          ; CMPP3 pack is DS-Standard, GPR0 only, no int. code needed.
;31959          ; Restart is at DS.CMPP.REE:.
;31960          ;
;31961          ; CMPP4 has 2 interrupt environments:
;31962          ; 1) During "seperate flows", which is the leading byte READs from the
;31963          ;    longer source.  This FPD pack/unpack is special and uses both
;31964          ;    GPR0 & GPR2.
;31965          ; 0) During common flows with CMPP3.  This FPD pack/unpack is
;31966          ;    DS-standard in GPR0, and special in GPR2.
;31967          ;
;31968          ; <ADDSUB>P<46> pack is DS-Standard: GPR0 and low word of GPR2
;31969          ; only.  No int. code is needed, restart is at DS.ADDP4.REE:.
;31970          ;
;31971          ; MULP pack is DS-Standard: GPR0 and low word of GPR2
;31972          ; only.  No int. code is needed, restart is at DS.ADDP4.REE:.
;31973          ;
;31974          ; If MTEMP8, MTEMP9, or (for <ADDSUB>Px and MULP ONLY) RTEMP8 unpacked
;31975          ; values are > 31., the GPR's have illegal data due to user meddling
;31976          ; and we ensure things by ANDing the bad value with 1F.
;31977          ;
;31978          ; GPR0 & GPR2 aren't zeroed after unpack because the packed values
;31979          ; are forced in fresh during each interrupt/BUS exception.
;31980          ;
;31981          ; In this banner, M=MOVP, C3=CMPP3, C4=CMPP4 in sep. flows, C6=CMPP4
;31982          ; when in CMPP3 code, +=<ADDSUB>P<46>, *=MULP, ^=ASHP.
;31983          ;
;31984          ;       Input           GPR0    #0:     FLAGS*(MC3C6+*^)
;31985          ;                                       TEMP3*(C4)
;31986          ;                               #1:     MTEMP8*(MC3C6+*^)
;31987          ;                                       TEMP1*(C4)
;31988          ;                               #2:     Zero, unused (int. code)
;31989          ;                                       (MC3C6+*^)
;31990          ;                                       1*(C4)
;31991          ;                               #3:     PC-PCBACK(MC3C4C6+*^)
;31992          ;                       GPR2    #0:     MTEMP9*(+*^)
;31993          ;                                       TEMP1*(C6)
;31994          ;                               #1:     RTEMP8(+*^)
;31995          ;                                       This is (*) for all except ASHP
;31996          ;                                       TEMP2*(C6)
;31997          ;                               #2:     RTEMP9(^)
;31998          ;                       <* appended to values means for byte it is >=0>
;31999          ;
;32000          ;       Resources       TEMP0           Restores PC (MC3C4C6+*^)
;32001          ;                       TEMP3           Temp. storage(+*^)
```

```
;32002          ;       Output          FLAGS           FLAGS(MC3C4C6+*^)
;32003          ;                       TEMP1           Source 1 length(C4)
;32004          ;                                       'Base address #1'' (C6)
;32005          ;                       TEMP2           'Base address #2'' (C6)
;32006          ;                       TEMP3           Source 2 length(C4)
;32007          ;                       MTEMP8          Source L (MTEMP8)(MC3C6+*^)
;32008          ;                       MTEMP9          Dest. L (MTEMP9)(^)
;32009          ;                                       Source 2 L (MTEMP9)(+*)
;32010          ;                       RTEMP8          Count (RTEMP8)(^)
;32011          ;                                       Dest. L (RTEMP8)(+*)
;32012          ;                       RTEMP9          Round operand (RTEMP9)(^)
;32013          ;                       GPR0            Untouched(MC3C4C6+*^)
;32014          ;                       GPR2            Untouched(MC3C4C6+*^)
;32015          ;                       TEMP7           Zero (0)(^)
;32016          ;                       STEPC           Source L/2(MC3C6)
;32017          ;                       DREG            Source L/2(MC3C6)
;32018          ;                       PC              Pointing past Istream
;32019          ;                                       (MC3C4C6+*^)
;32020          ;
;32021          ;       Note:   PLATCH_[8.] and FPDOFFSET_ZLITO[15.] at DS.ASHP.REE:.
;32022          ;               FPDOFFSET_ZLITO[17.] at DS.MOVP.REE:.
;32023          ;               FPDOFFSET_ZLITO[14.] at DS.ADDP4.REE:.
;32024          ;*****************************************************************************
```

```
                        ;32025 .REGION/IRD1.R1L,IRD1.R1H
                        ;32026 =000
                        ;32027 DS.DS.UNP:
                        ;32028          ;000--------------------------;
U 0380, 0880,05BE,4034,0307,013A,7      ;32029          FLAGS_R[R0]                 ;FLAGS
                        ;32030 =
                        ;32031
                        ;32032 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                        ;32033          ;--------------------------------;
U 13A7, 0487,929C,7024,0047,013A,8      ;32034          MTEMP0_R[R0].RR.24 Q_PCBACK ;TEMP0_deltaPC in byte#0, Q_PCBACK
                        ;32035
                        ;32036          ;--------------------------------;
U 13A8, 0480,0019,0000,0487,013A,9      ;32037          PC_ZEXT(M[TEMP0])+Q,SIZE[BYTE] ;PC is restored
                        ;32038
                        ;32039          ;--------------------------------;
                        ;32040          M[TEMP8]_R[R0].RR.8,        ;MTEMP8 data in low byte, MDR_Opcode
U 13A9, 0486,82B7,0004,0567,013A,A      ;32041          MDR_ZEXT(IR)
                        ;32042
                        ;32043          ;--------------------------------;
U 13AA, 0586,8C12,0030,F847,013A,B      ;32044          M[TEMP8]_MB.AND.ZLIT0[31.]  ;MTEMP8_packed value.  If the packed
                        ;32045                                      ;value was illegal, new value is a
                        ;32046                                      ;legal packed decimal length <= 31.
                        ;32047
                        ;32048          ;--------------------------------;
U 13AB, 0C81,2592,4230,0047,010A,6      ;32049          WB_M[MDR],WB<5-0>?          ;Yes, legal value, continue unpacking,
                        ;32050                                      ;which inst. are we?
                        ;32051
                        ;32052 =100110
                        ;32053          ;100110--------------------------;
                        ;32054          M[TEMP9]_R[R2],             ;ADDP4[20], SUBP4[22], get MTEMP9 data
U 10A6, 0C86,95BE,4034,8047,013B,2      ;32055          NEXT/DS.DS.UNP.2
                        ;32056
                        ;32057          ;100111--------------------------;
                        ;32058          M[TEMP9]_R[R2],             ;MULP[25], ADDP6[21], SUBP6[23], ditto
U 10A7, 0C86,95BE,4034,8047,013B,2      ;32059          NEXT/DS.DS.UNP.2
                        ;32060
                        ;32061 =110110
                        ;32062          ;110110--------------------------;
                        ;32063          STEPC_D_M[TEMP8].SR.1,      ;MOVP[34], STEPC_D_L/2 for DS.READ,
U 10B6, 0080,8001,A03D,8107,0100,D      ;32064          NEXT/DS.MOVP.REE            ;reenter inst.
                        ;32065
                        ;32066          ;110111--------------------------;
                        ;32067          STEPC_D_M[TEMP8].SR.1,IR<2-0>?, ;CMPP3[35], CMPP4[37], STEPC_DREG_
U 10B7, 0C80,8001,A67D,8107,0105,5      ;32068          NEXT/DS.DS.UNP.CMPP34       ;DS.READ data, which inst. are we?
                        ;32069
                        ;32070          ;111110--------------------------;
                        ;32071          M[TEMP9]_R[R2],             ;ASHP[F8], get MTEMP9 data
U 10BE, 0C86,95BE,4034,8047,013B,2      ;32072          NEXT/DS.DS.UNP.2
                        ;32073 =
```

```
                                ;32074          ;*****************************************************************************
                                ;32075          ; CMPP3 & CMPP4 split here. CMPP4 has 2 interrupt codes to decipher.
                                ;32076          ;*****************************************************************************
                                ;32077 =101
                                ;32078 DS.DS.UNP.CMPP34:
                                ;32079          ;101-----------------------------;
                                ;32080          M[FPDOFFSET]_ZLIT0[20.],        ;CMPP3[35], load FPDOFFSET, anything
                                ;32081          COUNT OR INT TIMER?,SIZE[LONG], ;pending?
U 1055, 0186,CC37,0B20,A047,0904,2 350*  ;32082          NEXT/DS.CMPP.REE
                                ;32083
                                ;32084          ;111-----------------------------;
U 1057, 0C80,02B7,02?4,0047,092B,2 337*  ;32085          WB_R[R0].RR.16,WB<0>?           ;CMPP4[37], which int. code are we?
                                ;32086
                                ;32087 =0
                                ;32088          ;0-------------------------------;
                                ;32089          M[TEMP1]_R[R2],                 ;CMPP4(#0), TEMP1_src.#1 offset data
U 12B2, 0086,15BE,4034,8047,013A,E       ;32090          NEXT/DS.DS.UNP.CMPP4.0
                                ;32091
                                ;32092          ;1-------------------------------;
U 12B3, 0C84,8592,4030,4047,013A,C       ;32093          R[TEMP1]_M[TEMP8]               ;CMPP4(#1), copy src.#1 length
                                ;32094
                                ;32095          ;-------------------------------;
U 13AC, 0086,35BE,4034,0047,013A,D       ;32096          M[TEMP3]_R[R0]                  ;TEMP3 data
                                ;32097
                                ;32098          ;-------------------------------;
                                ;32099          M[TEMP3]_MB.AND.ZLIT0[31.],     ;Legalized, reenter CMPP4, seperate
U 13AD, 0186,3C12,0030,F847,010E,9       ;32100          NEXT/DS.CMPP4.REE               ;flows
                                ;32101
                                ;32102
                                ;32103          ;*****************************************************************************
                                ;32104          ; CMPP4 interrupt code #0 continues here.  TEMP1 has source 1
                                ;32105          ; "base address" offset.  We don't mask out illegal packed offset
                                ;32106          ; values since it won't affect the instruction.
                                ;32107          ;*****************************************************************************
                                ;32108
                                ;32109 DS.DS.UNP.CMPP4.0:
                                ;32110          ;-------------------------------;
                                ;32111          M[TEMP1]_ZEXT(MB)+R[R1],        ;TEMP1_offset+base address
U 13AE, 0886,1015,0004,4047,013A,F       ;32112          SIZE[BYTE]
                                ;32113
                                ;32114          ;-------------------------------;
U 13AF, 0086,22B7,0004,8047,013B,0       ;32115          M[TEMP2]_R[R2].RR.8             ;TEMP2_source#2 offset
                                ;32116
                                ;32117          ;-------------------------------;
                                ;32118          M[TEMP2]_ZEXT(MB)+R[R3],        ;TEMP2_offset+base address
U 13B0, 0C86,2015,0004,C047,013B,1       ;32119          SIZE[BYTE]
                                ;32120
                                ;32121          ;-------------------------------;
                                ;32122          M[FPDOFFSET]_ZLIT0[20.],        ;Load FPDOFFSET, anything pending?
                                ;32123          COUNT OR INT TIMER?,SIZE[LONG],
U 13B1, 0186,CC37,0B20,A047,0904,2 350*  ;32124          NEXT/DS.CMPP.REE
```

```
                                ;32125          ;*****************************************************************************
                                ;32126          ; <ADDSUB>P<46>, MULP, and ASHP continue unpacking here
                                ;32127          ;*****************************************************************************
                                ;32128
                                ;32129 DS.DS.UNP.2:
                                ;32130          ;--------------------------------;
U 13B2, 0186,9C12,0030,F847,013B,3  ;32131      M[TEMP9]_MB.AND.ZLIT0[31.]      ;MTEMP9_packed data.  If illegal data
                                ;32132                                          ;was packed, MTEMP9_legal data <= 31.
                                ;32133
                                ;32134          ;--------------------------------;
U 13B3, 0C81,2592,4230,0047,010A,F  ;32135      WB_M[MDR],WB<5-0>?              ;Which inst. are we?
                                ;32136
                                ;32137 =101111
                                ;32138          ;101111----------------------------;
                                ;32139          M[TEMP3]_R[R2].RR.8,            ;ADDP4[20], ADDP6[21], SUBP4[22],
U 10AF, 0C86,32B7,0004,8047,013B,8  ;32140      NEXT/DS.DS.UNP.NASH             ;SUBP6[23], MULP[25],  get RTEMP8 data
                                ;32141
                                ;32142          ;111111----------------------------;
U 10BF, 0C86,32B7,0004,8047,013B,4  ;32143      M[TEMP3]_R[R2].RR.8             ;ASHP[F8], get RTEMP8 data
                                ;32144
                                ;32145
                                ;32146          ;*****************************************************************************
                                ;32147          ; ASHP continues unpacking here.
                                ;32148          ;*****************************************************************************
                                ;32149
                                ;32150          ;--------------------------------;
                                ;32151          R[TEMP8]_SEXT(M[TEMP3]),        ;RTEMP8 is unpacked
U 13B4, 0084,3E5E,0002,0047,013B,5  ;32152      SIZE[BYTE]
                                ;32153
                                ;32154          ;--------------------------------;
U 13B5, 0086,33B7,0000,0047,013B,6  ;32155      M[TEMP3]_MB.RR.8                ;RTEMP9 data in low byte
                                ;32156
                                ;32157          ;--------------------------------;
                                ;32158          R[TEMP9]_ZEXT(M[TEMP3]),        ;RTEMP9
U 13B6, 0C84,359E,4002,4047,013B,7  ;32159      SIZE[BYTE]
                                ;32160
                                ;32161          ;--------------------------------;
                                ;32162          M[TEMP7]_ZLIT0[0],              ;TEMP7_0, restart inst.
U 13B7, 0186,7C37,0030,0047,012A,2  ;32163      NEXT/DS.ASHP.REE
                                ;32164
                                ;32165
                                ;32166          ;*****************************************************************************
                                ;32167          ; ADDP4, ADDP6, MULP, SUBP4, and SUBP6 continue unpacking here.
                                ;32168          ;*****************************************************************************
                                ;32169
                                ;32170 DS.DS.UNP.NASH:
                                ;32171          ;--------------------------------;
U 13B8, 0186,3C12,0030,F847,013B,9  ;32172      M[TEMP3]_MB.AND.ZLIT0[31.]      ;RTEMP8 data is unpacked, if it is
                                ;32173                                          ;illegal, it is now legal
                                ;32174
                                ;32175          ;--------------------------------;
                                ;32176          R[TEMP8]_M[TEMP3],              ;Reenter ADDP4 code
U 13B9, 0084,3592,4032,0047,0111,2  ;32177      NEXT/DS.ADDP4.REE
```

```
                                ;32178 .TOC     "       Decimal String                  : DS.DIV10.DIVP"
                                ;32179
                                ;32180          ;*****************************************************************************
                                ;32181          ; DS.DIV10.DIVP divides a Packed Decimal number by 10.
                                ;32182          ;
                                ;32183          ;       Input           TEMP0-TEMP3     Packed Decimal data, memory
                                ;32184          ;                                       format, LSB in byte #3 of TEMP0,
                                ;32185          ;                                       MSB in byte #0 of TEMP3
                                ;32186          ;                       TEMP7           Zero (0)
                                ;32187          ;
                                ;32188          ;       Output          TEMP0-TEMP3     Old data divided by 10.
                                ;32189          ;                       TEMP7           Zero (0)
                                ;32190          ;
                                ;32191          ;       Returns         +1
                                ;32192          ;*****************************************************************************
                                ;32193 =0
                                ;32194 DS.DIV10.DIVP:
                                ;32195          ;0-----------------------------;
                                ;32196          PUSH,NEXT/DS.SR.123SWP,         ;BCDSWP TEMP0-TEMP3
U 12B4, 0C86,0637,0030,0047,053E,F ;32197       M[TEMP0]_MB.BCDSWP
                                ;32198
                                ;32199          ;1-----------------------------;
U 12B5, 0484,1277,0030,0047,013B,A ;32200       R[TEMP0]_(M[TEMP1] RB).RR.4     ;Divide by 10., starting with LSD
                                ;32201
                                ;32202          ;-----------------------------;
U 13BA, 0884,2277,0030,4047,013B,B ;32203       R[TEMP1]_(M[TEMP2] RB).RR.4
                                ;32204
                                ;32205          ;-----------------------------;
U 13BB, 0C84,3277,0030,8047,012B,6 ;32206       R[TEMP2]_(M[TEMP3] RB).RR.4
                                ;32207
                                ;32208 =0
                                ;32209          ;0-----------------------------;
                                ;32210          R[TEMP3]_(M[TEMP7] RB).RR.4,    ;Shift zero in high nibble, BCDSWP
U 12B6, 0C84,7277,0030,C047,053E,F ;32211       PUSH,NEXT/DS.SR.123SWP          ;TEMP1-3
                                ;32212
                                ;32213          ;1-----------------------------;
U 12B7, 0886,0637,00B0,0047,0000,1 ;32214       M[TEMP0]_MB.BCDSWP,RETURN [1]   ;All done, RETURN+1
```

```
                                ;32215 .TOC     "       Decimal String                  : DS.DIV10"
                                ;32216
                                ;32217          ;*************************************************************************
                                ;32218          ; DS.DIV10 divides a Packed Decimal number by 10.
                                ;32219          ;
                                ;32220          ;       Input           TEMP3-TEMP6     Packed Decimal data, memory
                                ;32221          ;                                       format, LSB in byte #3 of TEMP3,
                                ;32222          ;                                       MSB in byte #0 of TEMP6
                                ;32223          ;                       TEMP7           Zero (0)
                                ;32224          ;
                                ;32225          ;       Output          TEMP3-TEMP6     Old data divided by 10.
                                ;32226          ;                       TEMP7           Zero (0)
                                ;32227          ;
                                ;32228          ;       Returns         +2
                                ;32229          ;*************************************************************************
                                ;32230 =0
                                ;32231 DS.DIV10:
                                ;32232          ;0-----------------------------;
                                ;32233          PUSH,NEXT/DS.SR.456SWP,         ;BCDSWP TEMP3-TEMP6
U 12B8, 0C86,3637,0030,0047,053E,C ;32234       M[TEMP3]_MB.BCDSWP
                                ;32235
                                ;32236          ;1-----------------------------;
U 12B9, 0484,4277,0030,C047,013B,C ;32237       R[TEMP3]_(M[TEMP4] RB).RR.4     ;Divide by 10., starting with LSD
                                ;32238
                                ;32239          ;-----------------------------;
U 13BC, 0C84,5277,0031,0047,013B,D ;32240       R[TEMP4]_(M[TEMP5] RB).RR.4
                                ;32241
                                ;32242          ;-----------------------------;
U 13BD, 0884,6277,0031,4047,012B,A ;32243       R[TEMP5]_(M[TEMP6] RB).RR.4
                                ;32244
                                ;32245 =0
                                ;32246          ;0-----------------------------;
                                ;32247          R[TEMP6]_(M[TEMP7] RB).RR.4,    ;Shift zero in high nibble, BCDSWP
U 12BA, 0C84,7277,0031,8047,053E,C ;32248       PUSH,NEXT/DS.SR.456SWP          ;TEMP4-6
                                ;32249
                                ;32250          ;1-----------------------------;
U 12BB, 0886,3637,00B0,0047,0000,2 ;32251       M[TEMP3]_MB.BCDSWP,RETURN [2]   ;All done, RETURN+2
```

```
                                ;32252 .TOC     "       Decimal String                  : DS.MUL10"
                                ;32253
                                ;32254          ;*****************************************************************************
                                ;32255          ;
                                ;32256          ; DS.MUL10 multiplies a Packed Decimal number by 10.
                                ;32257          ;
                                ;32258          ;       Input           TEMP3-TEMP6     Packed Decimal data, memory
                                ;32259          ;                                       format, LSB in byte #3 of TEMP3,
                                ;32260          ;                                       MSB in byte #0 of TEMP6
                                ;32261          ;
                                ;32262          ;       Output          TEMP3-TEMP6     Old data multiplied by 10.
                                ;32263          ;
                                ;32264          ;       Returns         +2
                                ;32265          ;
                                ;32266          ;*****************************************************************************
                                ;32267 =0
                                ;32268 DS.MUL10:
                                ;32269          ;0-------------------------------;
                                ;32270          M[TEMP3]_MB.BCDSWP,PUSH,        ;BCDSWP TEMP3-TEMP6
U 12BC, 0C86,3637,0030,0047,053E,C      ;32271          NEXT/DS.SR.456SWP
                                ;32272
                                ;32273          ;1-------------------------------;
U 12BD, 0886,6237,0031,4047,013B,E      ;32274          M[TEMP6]_(R[TEMP5] MB).RL.4     ;Multiply by 10., starting with MSD
                                ;32275
                                ;32276          ;-------------------------------;
U 13BE, 0486,5237,0031,0047,013B,F      ;32277          M[TEMP5]_(R[TEMP4] MB).RL.4
                                ;32278
                                ;32279          ;-------------------------------;
U 13BF, 0486,4237,0030,C047,012B,E      ;32280          M[TEMP4]_(R[TEMP3] MB).RL.4
                                ;32281
                                ;32282 =0
                                ;32283          ;0-------------------------------;
                                ;32284          PUSH,NEXT/DS.SR.456SWP,         ;Shift zero in low nibble, BCDSWP
U 12BF, 0886,3237,003D,8047,053E,C      ;32285          M[TEMP3]_(R[ZERO] MB).RL.4      ;TEMP4-6
                                ;32286
                                ;32287          ;1-------------------------------;
U 12BF, 0886,363?,00B0,0047,0000,2      ;32288          M[TEMP3]_MB.BCDSWP,RETURN [2]   ;BCDSWP TEMP3, all done
```

```
;32289 .TOC     "       Decimal String                  : DS.READ"
;32290
;32291          ;*****************************************************************************
;32292          ; DS.READ has 3 entry points:
;32293          ; 1) DS.READ: to read a Packed Decimal string into TEMP3-TEMP6.
;32294          ; 2) DS.READ: on an IR<2-0>? BUT to read a Packed Decimal string into
;32295          ;    TEMP3-TEMP6.  =101 is CMPP3[35], and the READ takes place as if
;32296          ;    the routine was entered directly at DS.READ:.  =111 is CMPP4[37],
;32297          ;    and VA is reloaded with a different address, the length offset from
;32298          ;    the adjusted "base address".  This costs 1 extra cycle. Please
;32299          ;    read the banners on the CMPP4 and CMPP3 instructions.
;32300          ; 3) DS.READ.DIVP.DR: to read a Packed Decimal string into
;32301          ;    RTEMP8-RTEMP11 and perform the manipulation & digit counting that
;32302          ;    DIVP requires.
;32303          ;
;32304          ; In this banner "*" means for code which enters at DS.READ: and takes
;32305          ;                  either split (=101 or =111).
;32306          ;                  "%" means for code which enters at DS.READ.DIVP.DR:.
;32307          ;                  "$" means for code which enters at the =111 split off
;32308          ;                  DS.READ:.
;32309          ;                  nothing means for both.
;32310          ;
;32311          ;       Input           Intcheck        Made no more than
;32312          ;                                       <Max. loop>-27. cycles before
;32313          ;                                       calling routine.
;32314          ;                       VA              (Base address of string)+(L/2+1)
;32315          ;                       STEPC           L/2
;32316          ;                       DREG ($)        Source Length
;32317          ;                       TEMP1 ($)       Adjusted base address of source
;32318          ;                       FPDOFFSET       IANDE offset code
;32319          ;                       GPR0 (%)        deltaPC in byte#3
;32320          ;
;32321          ;       Resources       MDR             Used for READs
;32322          ;                       RNUM            Points to current TEMP
;32323          ;                       DREG            Copy of RNUM
;32324          ;                       FLAG0           Used to remember which mode
;32325          ;                                       we're in
;32326          ;                       TEMP0-TEMP3 (%) Used to read in source string
;32327          ;                       MTEMP8 (%)      Used to calc. DR(LDNLZ)
;32328          ;
;32329          ;       Output          TEMP3-TEMP6 (*) Packed Decimal data, memory
;32330          ;                                       format, LSB in byte#3 of TEMP3,
;32331          ;                                       MSB in byte#0 of TEMP6, sign
;32332          ;                                       nibble clear, unused (high)
;32333          ;                                       TEMPs zeroed
;32334          ;                       TEMP3           1 if (Bus. cycle excep. or
;32335          ;                                       Int.) AND (FPDOFFSET=0).  See
;32336          ;                                       below for special RETURN
;32337          ;                       RTEMP8-RTEMP11 (%)  Packed Decimal data, memory
;32338          ;                                       format, LSB in byte#3 of TEMP0,
;32339          ;                                       MSB in byte#0 of TEMP3, unused
;32340          ;                                       (high) TEMPs zero filled.  This
;32341          ;                                       is source divided by 10., sign
;32342          ;                                       nibble out, zero shifted in.
```

```
                                ;32343          ;                       FLAGO           =0 (*)
                                ;32344          ;                                       =1 (%)
                                ;32345          ;                       GPR2 (%)        DR(LDNLZ)
                                ;32346          ;                       Intcheck (*)    Made before RETURN..microword
                                ;32347          ;                                       on return is cycle #7
                                ;32348          ;                       Intcheck (%)    Made before RETURN..microword
                                ;32349          ;                                       on return is cycle #3
                                ;32350          ;                       PSL<V> (%)      -1 if DR=0
                                ;32351          ;                       FPD (%)         -0 if DR=0
                                ;32352          ;                       QREG (%)        -PCBACK if DR=0
                                ;32353          ;                       TEMP0 (%)       -PC if DR=0
                                ;32354          ;                       ALUS            BCDSIGN evaluation data
                                ;32355          ;
                                ;32356          ;               Returns         -1 (*)          Always
                                ;32357          ;                               +2              (Bus cycle excep. or int.) AND
                                ;32358          ;                                               (FPDOFFSET=0).  See output
                                ;32359          ;                                               section for TEMP3 listing.
                                ;32360          ;                               +3 (%)          Data read ok, DR<>0.
                                ;32361          ;                               +4 (%)          DR=0, see output section for
                                ;32362          ;                                               data on this return.
                                ;32363          ;
                                ;32364          ; Note that Decimal String uses this routine with FPDOFFSET=0 for
                                ;32365          ; DIVP, nonzero for all other instructions (MOVP,CMPPx,ADDPx,SUBPx,
                                ;32366          ; MULP,ASHP).
                                ;32367          ;*****************************************************************************
                                ;32368 =101
                                ;32369 DS.READ:
                                ;32370          ;101-----------------------------;
                                ;32371          RNUM_D_ZLITO[3],CLEAR FLAGO,    ;Low TEMP is TEMP3
 105D, 0501,0C55,2050,1847,0130,1 ;32372        NEXT/DS.READ.2
                                ;32373
                                ;32374          ;111-----------------------------;
 105F, 0080,00A1,0030,44A7,0105,0 ;32375        VA_D+R[TEMP1]+1,NEXT/DS.READ    ;**CMPP4 special entry, reload VA
                                ;32376
                                ;32377 DS.READ.DIVP.DR:
                                ;32378          ;-------------------------------;
 1300, 00A1,0C55,2030,0047,0130,1 ;32379        RNUM_D_ZLITO[0],SET FLAGO       ;Low TEMP is TEMP0
                                ;32380
                                ;32381
                                ;32382 DS.READ.2:
                                ;32383          ;-------------------------------;
                                ;32384          VA_M[VA]-ZLITO[4],              ;Pull back VA to lowest LONGWORD,
 1301, 0181,8C10,0370,24A7,0108,0 ;32385        STEPC.GE.4? DECBY4              ;how much is left?
```

```
                                ;32386          ;*************************************************************************
                                ;32387          ; STEPC BUT table.  =000 is 1 byte left in string, =001 is 1 word left
                                ;32388          ; in string, =010 is 3bytes left in string, =011 is 1 longword left in
                                ;32389          ; string, =100 and =101 are > 1 longword left in string, with something
                                ;32390          ; pending at =101 split.
                                ;32391          ;*************************************************************************
                                ;32392 =0000
                                ;32393 DS.READ.LOP:
                                ;32394          ;0000-------------------------;
                                ;32395          VA_M[VA]+ZLIT0[3],              ;Byte left, point VA
U 1080, 0981,BC11,0030,1CA7,0116,A ;32396       NEXT/DS.READ.BYT
                                ;32397
                                ;32398          ;0001-------------------------;
                                ;32399          VA_M[VA]+ZLIT0[2],              ;Word left, point VA
U 1081, 0D81,BC11,0030,14A7,0117,B ;32400       NEXT/DS.READ.WRD
                                ;32401
                                ;32402          ;0010-------------------------;
                                ;32403          VA_M[VA]+ZLIT0[2],              ;3 bytes left, point VA to low word
U 1082, 0D81,BC11,0030,14A7,011A,3 ;32404       NEXT/DS.READ.3B
                                ;32405
                                ;32406          ;0011-------------------------;
                                ;32407          READ,SIZE[LONG],                ;1 longword left
U 1083, 0080,0036,4020,0050,013C,D ;32408       NEXT/DS.READ.LONG
                                ;32409
                                ;32410          ;0100-------------------------;
                                ;32411          READ,SIZE[LONG],                ;Middle of string
U 1084, 0880,0036,4020,0050,013C,A ;32412       NEXT/DS.READ.MID
                                ;32413
                                ;32414          ;0101-------------------------;
                                ;32415          INTPEND OR TIMER?,PUSH,         ;Middle of string, something pending,
U 1085, 0480,0036,4AF0,0047,04F7,0 ;32416       NEXT/IE.SERV.IP.TS2             ;RETURN-1 or never
                                ;32417
                                ;32418 -0111
                                ;32419          ;0111-------------------------;
U 1087, 0586,3037,00B0,0847,0000,3 ;32420       M[TEMP3]_ZLIT0[1],RETURN [2]    ;BUS exception
                                ;32421
                                ;32422          ;1000-------------------------;
U 1088, 0586,3037,00B0,0847,0000,2 ;32423       M[TEMP3]_ZLIT0[1],RETURN [2]    ;BUS exception
                                ;32424
                                ;32425          ;1001-------------------------;
U 1089, 0586,3037,00B0,0847,0000,2 ;32426       M[TEMP3]_ZLIT0[1],RETURN [2]    ;Int. pending
                                ;32427 =
```

```
                        ;32428          ;*************************************************************************
                        ;32429          ; Continuation of Byte read
                        ;32430          ;*************************************************************************
                        ;32431 =0**
                        ;32432 DS.READ.BYT:
                        ;32433          ;0*----------------------------;
                        ;32434          READ,SIZE[BYTE],
U 116A, 0480,0036,4000,0050,013C,2      ;32435          NEXT/DS.READ.BYT.1
                        ;32436
                        ;32437          ;1*----------------------------;
U 116E, 0586,3C37,00B0,0847,0000,2      ;32438          M[TEMP3]_ZLIT0[1],RETURN [2]    ;BUS exception
                        ;32439
                        ;32440 DS.READ.BYT.1:
                        ;32441          ;-----------------------------;
U 1302, 0881,2016,400D,8467,013C,3      ;32442          MDR_ZEXT(M[MDR]),SIZE[BYTE]     ;Zero high 3 bytes
                        ;32443
                        ;32444          ;-----------------------------;
                        ;32445          R[TEMP,R]_M[MDR].RR.8,          ;Load TEMP, finish up
U 1303, 0085,2387,080C,18E7,0108,A      ;32446          NEXT/DS.READ.EX,IP.TS?
                        ;32447
                        ;32448
                        ;32449          ;*************************************************************************
                        ;32450          ; Continuation of Word read
                        ;32451          ;*************************************************************************
                        ;32452 =0**
                        ;32453 DS.READ.WRD:
                        ;32454          ;0*----------------------------;
                        ;32455          READ,SIZE[WORD],
U 1178, 0080,0036,4010,0050,013C,4      ;32456          NEXT/DS.READ.WRD.1
                        ;32457
                        ;32458          ;1*----------------------------;
U 117F, 0586,3C37,00B0,0847,0000,2      ;32459          M[TEMP3]_ZLIT0[1],RETURN [2]    ;BUS exception
                        ;32460
                        ;32461 DS.READ.WRD.1:
                        ;32462          ;-----------------------------;
U 1304, 0C81,2016,401D,8467,013C,5      ;32463          MDR_ZEXT(M[MDR]),SIZE[WORD]     ;Zero high word
                        ;32464
                        ;32465          ;-----------------------------;
                        ;32466          R[TEMP.R]_M[MDR].RR.16,         ;Load TEMP, finish up
U 1305, 0485,23B7,0B1C,18E7,0108,A      ;32467          NEXT/DS.READ.EX,IP.TS?
```

```
                                ;32468          ;*****************************************************************************
                                ;32469          ; Continuation of 3byte read
                                ;32470          ;*****************************************************************************
                                ;32471 =0**
                                ;32472 DS.READ.3B:
                                ;32473          ;0**-------------------------------;
                                ;32474          READ,SIZE[WORD],                ;Read low word
U 11A3, 0880,0036,4010,0050,013C,6 ;32475       NEXT/DS.READ.3B.1
                                ;32476
                                ;32477          ;1**-------------------------------;
U 11A7, 0586,3C37,00B0,0847,0000,2 ;32478       M[TEMP3]_ZLIT0[1],RETURN [2]    ;BUS exception, RETURN+2
                                ;32479
                                ;32480 DS.READ.3B.1:
                                ;32481          ;-------------------------------;
U 13C6, 0981,BC10,0030,0CA7,013C,7 ;32482       VA_M[VA]-ZLIT0[1]               ;Point VA to last byte
                                ;32483
                                ;32484          ;-------------------------------;
U 13C7, 0085,23B7,001C,0047,011B,8 ;32485       R[TEMP.R]_M[MDR].RR.16          ;Move data to high word
                                ;32486
                                ;32487 =0**
                                ;32488          ;0**-------------------------------;
                                ;32489          READ,SIZE[BYTE],                ;READ last byte, clear low word
                                ;32490          M[TEMP.R]_MB.CLR2B,             ;in TEMP
U 11B8, 0087,04B7,0000,0050,013C,8 ;32491       NEXT/DS.READ.3B.2
                                ;32492
                                ;32493          ;1**-------------------------------;
U 11BC, 0586,3C37,00B0,0847,0000,2 ;32494       M[TEMP3]_ZLIT0[1],RETURN [2]    ;BUS exception, RETURN+2
                                ;32495
                                ;32496 DS.READ.3B.2:
                                ;32497          ;-------------------------------;
U 13C8, 0881,2016,400D,8467,013C,9 ;32498       MDR_ZEXT(M[MDR]),SIZE[BYTE]     ;High 3 bytes_0
                                ;32499
                                ;32500          ;-------------------------------;
                                ;32501          R[TEMP.R]_RB.OR.(M[MDR].RR.24), ;TEMP has completed 3 bytes,
U 13C9, 0085,23BE,4B2C,18E7,0108,A ;32502       NEXT/DS.READ.EX,IP.TS?          ;finish up
                                ;32503
                                ;32504
                                ;32505          ;*****************************************************************************
                                ;32506          ; Continuation of longword READ, middle of string
                                ;32507          ;*****************************************************************************
                                ;32508
                                ;32509 DS.READ.MID:
                                ;32510          ;-------------------------------;
U 13CA, 0581,BC10,0030,24A7,013C,B ;32511       VA_M[VA]-ZLIT0[4]               ;Decrement VA
                                ;32512
                                ;32513          ;-------------------------------;
U 13CB, 0C85,2592,403C,0047,013C,C ;32514       R[TEMP.R]_M[MDR]                ;Get data
                                ;32515
                                ;32516          ;-------------------------------;
                                ;32517          RNUM_D_D+1,STEPC.GE.4? DECBY4,  ;Point to next TEMP, loop back
U 13CC, 0481,D0A1,237D,8047,0108,0 ;32518       NEXT7DS.READ.LOP
```

```
                                ;32519          ;*************************************************************************
                                ;32520          ; Continuation of longword read, end of string
                                ;32521          ;*************************************************************************
                                ;32522
                                ;32523 DS.READ.LONG:
                                ;32524          ;------------------------------;
U 13CD, 0085,2592,4B3C,18E7,0108,A ;32525       R[TEMP.R]_M[MDR],IP.TS?         ;Get data
                                ;32526
                                ;32527
                                ;32528          ;*************************************************************************
                                ;32529          ; Source read, enter on IP.TS? split.  After nothing or timer we split
                                ;32530          ; on which mode we're in.
                                ;32531          ;*************************************************************************
                                ;32532 =010
                                ;32533 DS.READ.EX:
                                ;32534          ;010-----------------------------;
                                ;32535          WB_M[TEMP3].BCDSWP,             ;Guess standard DS.READ, latch BCD sign
U 108A, 0C80,363E,443D,80C7,012C,0 ;32536       ALUS_BCD SIGN.ZERO,FLAG0?,      ;eval., which mode are we really?
                                ;32537          NEXT/DS.READ.EX.SPL
                                ;32538
                                ;32539          ;011-----------------------------;
U 108B, 0480,0036,4AF0,0047,04F7,0 ;32540       PUSH,INTPEND OR TIMER?,
                                ;32541          NEXT/IE.SERV.IP.TS2
                                ;32542
                                ;32543 =111
                                ;32544          ;111-----------------------------;
U 108F, 0586,3C37,00B0,0847,0000,2 ;32545       M[TEMP3]_ZLIT0[1],RETURN [2]    ;Int. pending, RETURN+2
                                ;32546
                                ;32547 =0
                                ;32548 DS.READ.EX.SPL:
                                ;32549          ;0-------------------------------;
                                ;32550          WB_D,WB<5-0>?,                  ;DS.READ, which TEMP are we in now?
U 12C0, 0880,0022,423D,8047,0111,0 ;32551       NEXT/DS.READ.EX.R               ;(DREG = 3, 4, 5, or 6)
                                ;32552
                                ;32553          ;1-------------------------------;
                                ;32554          WB_M[TEMP0].BCDSWP,             ;DS.READ.DIVP.DR, latch BCD sign eval.
                                ;32555          ALUS_BCD SIGN.ZERO,             ;from sign TEMP
U 12C1, 0880,063E,403D,80C7,013C,E ;32556       NEXT/DS.READ.EX.D
```

```
                                ;32557          ;*****************************************************************************
                                ;32558          ; Zero unused TEMPs, zero sign nibble & RETURN-1 (DS.READ)
                                ;32559          ;*****************************************************************************
                                ;32560 =000
                                ;32561 DS.READ.EX.R:
                                ;32562 =011
                                ;32563          ;011-----------------------------;
U 1113, 0986,4C37,0030,0047,0111,4  ;32564          M[TEMP4]_ZLIT0[0]               ;3, zero TEMP4, TEMP5, and TEMP6
                                ;32565
                                ;32566          ;100-----------------------------;
U 1114, 0586,5C37,0030,0047,0111,5  ;32567          M[TEMP5]_ZLIT0[0]               ;(4, zero TEMP5 and TEMP6)
                                ;32568
                                ;32569          ;101-----------------------------;
U 1115, 0586,6C37,0030,0047,0111,6  ;32570          M[TEMP6]_ZLIT0[0]               ;(5, zero TEMP6)
                                ;32571
                                ;32572          ;110-----------------------------;
                                ;32573          M[TEMP3]_MB.ANDNOT.ZLIT24[0F],  ;(6, TEMPs are full, do sign eval)
U 1116, 0586,3C93,80B0,7847,03FF,F  ;32574          RETURN [-1]                     ;OR--> we zeroed high TEMPs, eval. sign
                                ;32575 =
                                ;32576
                                ;32577
                                ;32578          ;*****************************************************************************
                                ;32579          ; DS.READ.DIVP.DR mode continues here.  We clear unused (high) TEMPs,
                                ;32580          ; divide the string by 10. to shift out sign nibble, and count the
                                ;32581          ; number of significant digits & store the result in GPR2.
                                ;32582          ;*****************************************************************************
                                ;32583
                                ;32584 DS.READ.EX.D:
                                ;32585          ;-------------------------------;
U 13CE, 0480,0022,427D,8047,0120,0  ;32586          WB_D,WB<1-0>?                   ;Which TEMP are we pointing to now?
                                ;32587
                                ;32588 =00
                                ;32589          ;00------------------------------;
                                ;32590          M[TEMP8]_ZLIT0[8.],             ;TEMP0, Maximum of 8 digits in this
U 1200, 0586,8C37,0030,4047,013C,F  ;32591          NEXT/DS.READ.EX.D.0             ;string
                                ;32592
                                ;32593          ;01------------------------------;
                                ;32594          M[TEMP8]_ZLIT0[16.],            ;TEMP1, Max. of 16. digits
U 1201, 0D86,8C37,0030,8047,013D,0  ;32595          NEXT/DS.READ.EX.D.1
                                ;32596
                                ;32597          ;10------------------------------;
                                ;32598          M[TEMP8]_ZLIT0[24.],            ;TEMP2, Max. of 24. digits
U 1202, 0186,8C37,0030,C047,013D,1  ;32599          NEXT/DS.READ.EX.D.2
                                ;32600
                                ;32601          ;11------------------------------;
                                ;32602          M[TEMP8]_ZLIT0[32.],            ;TEMP3, Max. of 32. digits (reall  only
U 1203, 0586,8C37,0031,0047,012C,2  ;32603          NEXT/DS.READ.EX.D.SET           ;31., but zero will be shifted in by
                                ;32604                                          ;divide)
```

```
                                ;32605 DS.READ.EX.D.0:
                                ;32606          ;--------------------------------;
U 13CF, 0986,1C37,0030,0047,013D,0  ;32607          M[TEMP1]_ZLIT0[0]               ;Zero high 3 TEMPs
                                ;32608
                                ;32609 DS.READ.EX.D.1:
                                ;32610          ;--------------------------------;
U 13D0, 0186,2C37,0030,0047,013D,1  ;32611          M[TEMP2]_ZLIT0[0]               ;(Zero high 2 TEMPs)
                                ;32612
                                ;32613 DS.READ.EX.D.2:
                                ;32614          ;--------------------------------;
U 13D1, 0586,3C37,0030,0047,012C,2  ;32615          M[TEMP3]_ZLIT0[0]               ;(Zero high 1 TEMP)
                                ;32616
                                ;32617          ;*****************************************************************************
                                ;32618          ; High TEMPs are zero, RNUM & DREG points to current TEMP (0-3),
                                ;32619          ; and MTEMP8 contains the maximum number of significant digits.
                                ;32620          ;*****************************************************************************
                                ;32621 =0
                                ;32622 DS.READ.EX.D.SET:
                                ;32623          ;0--------------------------------;
                                ;32624          M[TEMP7]_ZLIT0[0],PUSH,         ;Load zero for shift-in, divide string
U 12C2, 0186,7C37,0030,0047,052B,4  ;32625          NEXT/DS.DIV10.DIVP              ;by 10. to shift out sign nibble
                                ;32626
                                ;32627          ;1--------------------------------;
                                ;32628          WB_M[TEMP.R].AND.ZLIT0[OFF],    ;Is high byte of current TEMP
U 12C3, 0D81,0C12,0A77,F847,012C,4  ;32629          WX.NE.0?                        ;nonzero?
                                ;32630
                                ;32631 =0
                                ;32632 DS.READ.EX.D.LOP:
                                ;32633          ;0--------------------------------;
                                ;32634          WB_M[TEMP.R].AND.ZLIT8[OFF],    ;No, is byte#1 nonzero?
U 12C4, 0581,0D92,0A77,F847,012C,6  ;32635          NEXT/DS.READ.EX.D.LOP1,WX.NE.0?
                                ;32636
                                ;32637          ;1--------------------------------;
                                ;32638          WB_M[TEMP.R].AND.ZLIT0[OF0],    ;Yes, is the high nibble of this byte
U 12C5, 0981,0C12,0A77,8047,012C,C  ;32639          WX.NE.0?,NEXT/DS.READ.EX.D.EX   ;nonzero?
                                ;32640
                                ;32641 =0
                                ;32642 DS.READ.EX.D.LOP1:
                                ;32643          ;0--------------------------------;
                                ;32644          WB_M[TEMP.R].AND.ZLIT16[OFF],   ;No, is byte#2 nonzero?
U 12C6, 0981,0D12,0A77,F847,012C,8  ;32645          WX.NE.0?,NEXT/DS.READ.EX.D.LOP2
                                ;32646
                                ;32647          ;1--------------------------------;
                                ;32648          M[TEMP8]_MB-ZLIT0[2],           ;Yes, max. sig. digit is in byte#1,
U 12C7, 0586,8C10,0030,1047,013D,4  ;32649          NEXT/DS.READ.EX.D.LT8           ;dec. counter by 2
                                ;32650
                                ;32651 =0
                                ;32652 DS.READ.EX.D.LOP2:
                                ;32653          ;0--------------------------------;
                                ;32654          WB_M[TEMP.R].AND.ZLIT24[OFF],   ;No, is the lowest byte nonzero?
U 12C8, 0181,0C92,0A77,F847,012C,A  ;32655          WX.NE.0?,NEXT/DS.READ.EX.D.LOP3
                                ;32656
                                ;32657          ;1--------------------------------;
                                ;32658          M[TEMP8]_MB-ZLIT0[4],           ;Yes, max. sig. digit is in byte#2,
U 12C9, 0186,8C10,0030,2047,013D,5  ;32659          NEXT/DS.READ.EX.D.LT16          ;dec. counter by 4
```

```
                                ;32660 =0
                                ;32661 DS.READ.EX.D.LOP3:
                                ;32662          ;0------------------------------;
                                ;32663          M[TEMP8]_MB-ZLIT0[8.],          ;No, this TEMP was zero, dec. counter
U 12CA, 0D86,8C10,0030,4047,013D,3  ;32664      NEXT/DS.READ.EX.D.LOP.END       ;by a LONGWORDs worth
                                ;32665
                                ;32666          ;1------------------------------;
U 12CB, 0D86,8C10,0030,3047,013D,2  ;32667      M[TEMP8]_MB-ZLIT0[6.]           ;Yes, adjust counter
                                ;32668
                                ;32669          ;------------------------------;
                                ;32670          WB_M[TEMP.R].AND.ZLIT24[0F0],   ;Is the high nibble of this byte
U 13D2, 0D81,0C92,0A77,8047,012C,C  ;32671      WX.NE.0?,NEXT/DS.READ.EX.D.EX   ;nonzero?
                                ;32672
                                ;32673
                                ;32674          ;*****************************************************************************
                                ;32675          ; Current TEMP is zero & TEMP8 has been decremented by 8.  If no more
                                ;32676          ; TEMPs exist, return for DR=0, else loop back to check next lowest
                                ;32677          ; TEMP.
                                ;32678          ;*****************************************************************************
                                ;32679
                                ;32680 DS.READ.EX.D.LOP.END:
                                ;32681          ;------------------------------;
U 13D3, 0481,D0A0,26FD,8047,0920,4 344*  ;32682  RNUM_D_D-1,WB<31-30>?           ;Are we negative?
                                ;32683
                                ;32684 =00
                                ;32685          ;00-----------------------------;
                                ;32686          WB_M[TEMP.R].AND.ZLIT0[0FF],    ;No, is byte#0 of this TEMP nonzero?
U 1204, 0D81,0C12,0A77,F847,012C,4  ;32687      WX.NE.0?,NEXT/DS.READ.EX.D.LOP  ;(loop thru this TEMP)
                                ;32688
                                ;32689 =11
                                ;32690          ;11-----------------------------;
                                ;32691          SET V,CLEAR FPD,RETURN [4],     ;Yes, all TEMPs were zero, take
U 1207, 0CE7,929C,70A4,08E7,0000,4  ;32692      MTEMP0_R[R0].RR.24 Q_PCBACK     ;DIVP divisor=0 RETURN+4
                                ;32693
                                ;32694
                                ;32695          ;*****************************************************************************
                                ;32696          ; Continuation of some BUTs in the DS.READ.EX.D.LOPx sequence.
                                ;32697          ;*****************************************************************************
                                ;32698
                                ;32699 DS.READ.EX.D.LT8:
                                ;32700          ;------------------------------;
                                ;32701          WB_M[TEMP.R].AND.ZLIT8[0F0],    ;Is nibble #3 nonzero?
U 13D4, 0981,0D92,0A77,8047,012C,C  ;32702      WX.NE.0?,NEXT/DS.READ.EX.D.EX
                                ;32703
                                ;32704 DS.READ.EX.D.LT16:
                                ;32705          ;------------------------------;
                                ;32706          WB_M[TEMP.R].AND.ZLIT16[0F0],   ;Is nibble #5 nonzero?
U 13D5, 0D81,0D12,0A77,8047,012C,C  ;32707      WX.NE.0?
```

```
                                ;32708          ;*************************************************************************
                                ;32709          ; The highest nonzero byte has been discovered.
                                ;32710          ; MTEMP8 contains the number of most significant digits if the high
                                ;32711          ;       nibble of nonzero byte is nonzero (ie the "most most").
                                ;32712          ; =0 is high nibble of current byte is zero, the MSD is in the low
                                ;32713          ;       nibble of this byte, GPR2_MTEMP8-1.
                                ;32714          ; =1 is high nibble of current byte is nonzero, GPR2_MTEMP8.
                                ;32715          ;*************************************************************************
                                ;32716 =0
                                ;32717 DS.READ.EX.D.EX:
                                ;32718          ;0-----------------------------;
                                ;32719          R[R2]_M[TEMP8]-1,
U 12CC, 0084,8E51,0034,8047,013D,6  ;32720      NEXT/DS.READ.EX.D.EX.F
                                ;32721
                                ;32722          ;1-----------------------------;
U 12CD, 0084,8592,4034,8047,013D,6  ;32723      R[R2]_M[TEMP8]
                                ;32724
                                ;32725 DS.READ.EX.D.EX.F:
                                ;32726          ;-----------------------------;
U 13D6, 0884,0592,4032,0047,013D,7  ;32727      R[TEMP8]_M[TEMP0]
                                ;32728
                                ;32729          ;-----------------------------;
U 13D7, 0884,1592,4032,4047,013D,8  ;32730      R[TEMP9]_M[TEMP1]
                                ;32731
                                ;32732          ;-----------------------------;
U 13D8, 0484,2592,4B32,98E7,0111,8  ;32733      R[TEMP10]_M[TEMP2],IP.TS?       ;Anything pending?
                                ;32734
                                ;32735 =000
                                ;32736          ;000-----------------------------;
                                ;32737          R[TEMP11]_M[TEMP3],             ;No, or timer service
U 1118, 0484,3592,40B2,C047,0000,3  ;32738      RETURN [3]
                                ;32739
                                ;32740          ;001-----------------------------;
                                ;32741          PUSH,INTPEND OR TIMER?,         ;Yes, RETURN -1 or +4
U 1119, 0480,0036,4AF0,0047,04F7,0  ;32742      NEXT/IE.SERV.IP.TS2
                                ;32743
                                ;32744 =101
                                ;32745          ;101-----------------------------;
                                ;32746          M[TEMP3]_ZLIT0[1],              ;Int. pending
U 111D, 0586,3C37,00B0,0847,0000,2  ;32747      RETURN [2]
                                ;32748 =
```

```
;32749 .TOC     "       Decimal String                  : DS.WRITE"
;32750
;32751          ;*************************************************************************
;32752          ; DS.WRITE writes a Packed Decimal string from TEMP3-TEMP6.
;32753          ;
;32754          ;       Input           TEMP3-TEMP6     DS data, memory format, LSB in
;32755          ;                                       byte#3 of TEMP3, MSB in byte#0
;32756          ;                                       TEMP6
;32757          ;                       FPDOFFSET       Nonzero
;32758          ;                       Intcheck        Made no more than
;32759          ;                                       <Max. loop>-8. cycles before
;32760          ;                                       calling routine.
;32761          ;                       VA              Base address of string
;32762          ;                       STEPC           L/2
;32763          ;                       QREG            L/2
;32764          ;                       PSL<C>          =0 if dest. is even
;32765          ;                                       =1 if dest. is odd
;32766          ;                       FLAG1           <V> image
;32767          ;                       FLAG2           <Z> image, =1 before call
;32768          ;
;32769          ;       Resources       FLAG0           Loop control
;32770          ;                       WDR             Used in WRITEs
;32771          ;                       RNUM            Points to current TEMP
;32772          ;                       DREG            Copy of RNUM
;32773          ;                       TEMP7           Points VA after PROBE
;32774          ;
;32775          ;       Output          Destination     Least signif. digits written,
;32776          ;                                       high nibble clear if PSL<C>=0
;32777          ;                       Intcheck        Made before first write..
;32778          ;                                       microword on return is cycle
;32779          ;                                       #58.
;32780          ;                       FLAG0           _0
;32781          ;                       FLAG1           _1 if overflow occured
;32782          ;                       FLAG2           _0 if nonzero written to memory
;32783          ;                       TEMP3-TEMP6     Garbage
;32784          ;
;32785          ;       Returns         -1              Always
;32786          ;*************************************************************************
```

```
                                ;32787 =0
                                ;32788 DS.WRITE:
                                ;32789          ;0------------------------------;
                                ;32790          RNUM_D_ZLIT0[3],SET FLAG0,      ;Point RNUM & DREG to first TEMP,
U 12CE, 0541,DC35,2030,1847,0579,A ;32791       PUSH,NEXT/MM.PRB.WRITE.NR       ;PROBE lowest accessed byte
                                ;32792
                                ;32793          ;1------------------------------;
U 12CF, 0885,B089,0B31,D8E7,012D,0 ;32794       R[TEMP7]_M[VA]+Q+1,IP.TS?       ;Calc. sign byte +1
                                ;32795
                                ;32796
                                ;32797          ;*********************************************************************
                                ;32798          ; Top of DS.WRITE loop.  No int. checks occur after the first check
                                ;32799          ; since WRITE must be completed for ADDP4-SUBP4.  Note that we have
                                ;32800          ; probed the last accessed byte of the dest. so if an exception or TB
                                ;32801          ; miss occurs, it will be during the probe & nothing will have been
                                ;32802          ; written.
                                ;32803          ;*********************************************************************
                                ;32804          ;
                                ;32805          ;+++++++++++++++++++++++++++++
                                ;32806          ;WE ARE GOING TO SET MM.NOINT ANYWAY - PROBE CAN'T BE TRUSTED
                                ;32807          ;+++++++++++++++++++++++++++++
                                ;32808
                                ;32809 =0
                                ;32810          ;0------------------------------;
                                ;32811
                                ;32812          SET MM.NOINT,                   ;WE WON'T ALLOW INTERRUPTS ANYWAY
                                ;32813          VA_M[TEMP7]-ZLIT0[4],           ;Point VA to first
                                ;32814          STEPC.GE.4? DECBY4,             ;longword, how much is left?
U 12D0, 0560,7C10,0370,24A7,0112,0 ;32815       NEXT/DS.WRITE.LOP.1
                                ;32816
                                ;32817          ;1------------------------------;
                                ;32818          PUSH,INTPEND OR TIMER?,
U 12D1, 0480,0036,4AF0,0047,04F7,0 ;32819       NEXT/IE.SERV.IP.TS2
```

```
                                ;32820 DS.WRITE.LOP:
                                ;32821          ;-------------------------------;
                                ;32822          VA_M[VA]-ZLIT0[4],              ;Point VA to next LONGWORD,
U 13D9, 0981,BC10,0370,24A7,0112,0  ;32823      STEPC.GE.4? DECBY4              ;longword, how much is left?
                                ;32824
                                ;32825 =000
                                ;32826 DS.WRITE.LOP.1:
                                ;32827          ;000----------------------------;
                                ;32828          VA_M[VA]+ZLIT0[3],PSL<C>?,      ;BYTE left, point VA, is dest. odd?
U 1120, 0581,BC11,0970,1CA7,012D,6  ;32829      NEXT/DS.WRITE.BYT
                                ;32830
                                ;32831          ;001----------------------------;
                                ;32832          VA_M[VA]+ZLIT0[2],PSL<C>?,      ;WORD left, point VA, ditto.
U 1121, 0181,BC11,0970,14A7,012E,0  ;32833      NEXT/DS.WRITE.WRD
                                ;32834
                                ;32835          ;010----------------------------;
                                ;32836          VA_M[VA]+ZLIT0[3],FLAG0?,       ;3BYTES left, point VA to low byte,
U 1122, 0D81,BC11,0430,1CA7,012E,8  ;32837      NEXT/DS.WRITE.3B                ;is this first longword?
                                ;32838
                                ;32839          ;011----------------------------;
U 1123, 0880,0036,4970,0047,012E,C  ;32840      PSL<C>?,NEXT/DS.WRITE.LNG       ;LONGWORD left, is dest. odd?
                                ;32841
                                ;32842          ;100----------------------------;
                                ;32843          WRITE M[TEMP.R],SIZE[LONG],     ;> LONGWORD left, WRITE it, is this
U 1124, 0481,0592,4420,05D8,012D,2  ;32844      FLAG0?,NEXT/DS.WRITE.MID        ;first longword?
                                ;32845
                                ;32846          ;101----------------------------;
                                ;32847          WRITE M[TEMP.R],SIZE[LONG],     ;> LONGWORD left, WRITE it, is this
U 1125, 0481,0592,4420,05D8,012D,2  ;32848      FLAG0?                          ;first longword?
                                ;32849 =
```

```
                                ;32850          ;*****************************************************************************
                                ;32851          ; Continuation, middle of string.  =0 is not first longword, =1 is
                                ;32852          ; first longword.
                                ;32853          ;*****************************************************************************
                                ;32854 =0
                                ;32855 DS.WRITE.MID:
                                ;32856          ;0-----------------------------;
                                ;32857          WB_M[TEMP.R],WX.NE.0?,          ;Is dest. nonzero?
U 12D2, 0081,0592,4A70,0047,012D 4 ;32858       NEXT/DS.WRITE.MID.2
                                ;32859
                                ;32860          ;1-----------------------------;
                                ;32861          WB_M[TEMP.R].ANDNOT.ZLIT24[0F], ;Ditto, without sign nibble
U 12D3, 0901,0C93,8A70,7847,012D,4 ;32862       WX.NE.0?,CLEAR FLAG0
                                ;32863
                                ;32864 =0
                                ;32865 DS.WRITE.MID.2:
                                ;32866          ;0-----------------------------;
U 12D4, 0081,D0A1,203D,8047,013D,9 ;32867       RNUM_D_D+1,NEXT/DS.WRITE.LOP    ;No, inc. to next TEMP, loop back
                                ;32868
                                ;32869          ;1-----------------------------;
                                ;32870          RNUM_D_D+1,CLEAR FLAG2,         ;Yes, inc. pointers, clear <Z> image,
U 12D5, 0011,D0A1,203D,8047,013D,9 ;32871       NEXT/DS.WRITE.LOP               ;loop back
```

```
                        ;32872          ;*************************************************************************
                        ;32873          ; Continuation, BYTE write.  =0 is even dest., =1 is odd dest.
                        ;32874          ;*************************************************************************
                        ;32875 =0
                        ;32876 DS.WRITE.BYT:
                        ;32877          ;0-----------------------------;
                        ;32878          WB_M[TEMP.R].ANDNOT.ZLIT24[0F], ;Do we have overflow, even dest.?
                        ;32879          NEXT/DS.WRITE.BYT.EV,
12D6, 0D81,0C93,8A30,7847,012D,A        ;32880          WX.EQ.0?
                        ;32881
                        ;32882          ;1-----------------------------;
                        ;32883          WB_M[TEMP.R].ANDNOT.ZLIT24[OFF],;Overflow, odd dest.?
12D7, 0181,0C93,8A77,F847,012D,8        ;32884          WX.NE.0?
                        ;32885
                        ;32886 =0
                        ;32887          ;0-----------------------------;
                        ;32888          R[TEMP.R]_RB.RR.24,             ;No, get byte into position, is this
12D8, 0884,02B7,042C,0047,012D,C        ;32889          FLAG0?,NEXT/DS.WRITE.BYT.EX     ;first WRITE?
                        ;32890
                        ;32891          ;1-----------------------------;
                        ;32892          SET FLAG1,FLAG0?,               ;Yes, set <V> image, is this first BYTE,
                        ;32893          R[TEMP.R]_RB.RR.24,             ;and get it into position
12D9, 004C,02B7,042C,0047,012D,C        ;32894          NEXT/DS.WRITE.BYT.EX
                        ;32895
                        ;32896          ;*************************************************************************
                        ;32897          ; Even string, =1 is no ov, =0 is ov
                        ;32898          ;*************************************************************************
                        ;32899 =0
                        ;32900 DS.WRITE.BYT.EV:
                        ;32901          ;0-----------------------------;
                        ;32902          SET FLAG1,                      ;<V> image_1, zero high nibble
12DA, C14F,0C93,8032,8047,012D,B        ;32903          M[TEMP.R]_MB.ANDNOT.ZLIT24[0F0]
                        ;32904
                        ;32905          ;1-----------------------------;
                        ;32906          R[TEMP.R]_RB.RR.24,             ;(And) get BYTE into position, is this
12DB, 0884,02B7,042C,0047,012D,C        ;32907          FLAG0?                          ;first WRITE?
```

```
                        ;32908          ;*****************************************************************************
                        ;32909          ; From even and odd BYTE paths, =0 is last WRITE, =1 is first WRITE,
                        ;32910          ; and mask out sign nibble accordingly when checking for zero to dest.
                        ;32911          ;*****************************************************************************
                        ;32912 =0
                        ;32913 DS.WRITE.BYT.EX:
                        ;32914          ;0-----------------------------;
                        ;32915          WB_M[TEMP.R].AND.ZL!TO[OFF],    ;Is byte nonzero?
12DC, 0581,0C12,0A77,F847,012D,E        ;32916          WX.NE.0?,NEXT/DS.WRITE.BYT.OUT
                        ;32917
                        ;32918          ;1-----------------------------;
                        ;32919          WB_M[TEMP.R].AND.ZL!TO[OFO],    ;Ditto, without sign nibble
12DD, 0101,0C12,0A77,8047,012D,E        ;32920          WX.NE.0?,CLEAR FLAG0
                        ;32921
                        ;32922 =0
                        ;32923 DS.WRITE.BYT.OUT:
                        ;32924          ;0-----------------------------;
                        ;32925          WRITE M[TEMP.R],SIZE[BYTE],     ;Byte is zero, WRITE & exit
12DE, 0C81,0592,4000,05D8,012F,2        ;32926          NEXT/DS.WRITE.EXIT
                        ;32927
                        ;32928          ;1-----------------------------;
                        ;32929          WRITE M[TEMP.R],SIZE[BYTE],     ;Byte is nonzero
12DF, 0C11,0592,4000,05D8,012F,2        ;32930          CLEAR FLAG2,NEXT/DS.WRITE.EXIT
```

```
                                ;32931          ;*****************************************************************************
                                ;32932          ; Continuation, WORD WRITE, =0 is even string, =1 is odd string.
                                ;32933          ;*****************************************************************************
                                ;32934 =0
                                ;32935 DS.WRITE.WRD:
                                ;32936          ;0-------------------------------;
                                ;32937          WB_M[TEMP.R].AND.ZLIT16[0F0],   ;Is high nibble zero?
12EC, 0DB1,0D13,0A77,8047,012E,2 ;32938         WX.NE.0?,NEXT/DS.WRITE.WRD.EV
                                ;32939
                                ;32940          ;1-------------------------------;
                                ;32941          WB_M[TEMP.R]-(MB.CLR2B),        ;Is extra word nonzero?
12E1, 0C81,0490,0A70,0047,092E,4 375* ;32942    WX.NE.0?,NEXT/DS.WRITE.WRD.OUT
                                ;32943
                                ;32944 =0
                                ;32945 DS.WRITE.WRD.EV:
                                ;32946          ;0-------------------------------;
                                ;32947          WB_M[TEMP.R]-(MB.CLR2B),        ;Even string, high nibble ok,
12E2, 0C81,0490,0A70,0047,092E,4 375* ;32948    WX.NE.0?,NEXT/DS.WRITE.WRD.OUT  ;is extra word ok?
                                ;32949
                                ;32950          ;1-------------------------------;
                                ;32951          SET FLAG1,                      ;High nibble nonzero in even string,
12E3, 014F,0D13,8037,8047,012E,4 ;32952         M[TEMP.R]_MB.ANDNOT.ZLIT16[0F0] ;<V> image_1, zero high nibble, don't
                                ;32953                                          ;check extra word as we have OV
                                ;32954
                                ;32955
                                ;32956          ;*****************************************************************************
                                ;32957          ; Here on BUT or to DS.WRITE.WRD.OUT: directly if OV occured in even
                                ;32958          ; string.  On BUT, =0 is ok, =1 is high word is nonzero, <V>image_1.
                                ;32959          ;*****************************************************************************
                                ;32960 =0
                                ;32961 DS.WRITE.WRD.OUT:
                                ;32962          ;0-------------------------------;
                                ;32963          WB_M[TEMP.R].RR.16,             ;Get data in low word, WRITE it,
                                ;32964          WRITE,SIZE[WORD],FLAG0?,        ;is this first WRITE?
12F4, 0C81,03B7,0410,0508,012F,6 ;32965         NEXT/DS.WRITE.WRD.EX
                                ;32966
                                ;32967          ;1-------------------------------;
                                ;32968          WB_M[TEMP.R].RR.16,             ;Ditto, we have OV
                                ;32969          WRITE,SIZE[WORD],FLAG0?,
12F5, 0449,03B7,0410,0508,012E,6 ;32970         SET FLAG1,NEXT/DS.WRITE.WRD.EX
                                ;32971
                                ;32972          ;*****************************************************************************
                                ;32973          ; =0 is last WRITE, =1 is first WRITE
                                ;32974          ;*****************************************************************************
                                ;32975 =0
                                ;32976 DS.WRITE.WRD.EX:
                                ;32977          ;0-------------------------------;
                                ;32978          WB_M[TEMP.R].CLR2B,             ;Clear high word, did dest._zero?
12E6, 0C81 04B7,0A70,0047,012F,2 ;32979         WX.NE.0?,NEXT/DS.WRITE.EXIT
                                ;32980
                                ;32981          ;1-------------------------------;
                                ;32982          CLEAR FLAG0,                    ;Clear sign nibble for zero to dest.
                                ;32983          M[TEMP.R]_MB.ANDNOT.ZLIT24[0F], ;check
12F7, 0507,0C93,8030,7847,012E,6 ;32984         NEXT/DS.WRITE.WRD.EX
```

```
                                ;32985          ;*****************************************************************************
                                ;32986          ; Continuation of 3BYTE WRITE, =0 is last WRITE, =1 is first WRITE.
                                ;32987          ;*****************************************************************************
                                ;32988 =0
                                ;32989 DS.WRITE.3B:
                                ;32990          ;0-------------------------------;
                                ;32991          WB_M[TEMP.R].AND.ZLIT24[0FF],   ;No sign, is byte nonzero?
U 12E8, 0981,0C92,0A77,F847,012E,A ;32992       WX.NE.0?,NEXT/DS.WRITE.3B.NZ
                                ;32993
                                ;32994          ;1-------------------------------;
                                ;32995          CLEAR FLAG0,WX.NE.0?,           ;FLAG0_0, is high nibble nonzero?
U 12E9, 0D01,0C92,0A77,8047,012E,A ;32996       WB_M[TEMP.R].AND.ZLIT24[0F0]
                                ;32997
                                ;32998 =0
                                ;32999 DS.WRITE.3B.NZ:
                                ;33000          ;0-------------------------------;
                                ;33001          M[TEMP.R]_MB.RR.24,             ;Yes, get BYTE into position
U 12EA, 0087,03B7,0020,0047,013D,A ;33002       NEXT/DS.WRITE.3B.OUT
                                ;33003
                                ;33004          ;1-------------------------------;
                                ;33005          CLEAR FLAG2,                    ;<2>image_0, get BYTE into pos.
U 12EB, 0017,03B7,0020,0047,013D,A ;33006       M[TEMP.R]_MB.RR.24
                                ;33007
                                ;33008
                                ;33009 DS.WRITE.3B.OUT:
                                ;33010          ;-------------------------------;
U 13DA, 0C81,0592,4000,05D8,013D,B ;33011       WRITE M[TEMP.R],SIZE[BYTE]      ;WRITE low BYTE
                                ;33012
                                ;33013          ;-------------------------------;
U 13DB, 0487,04F7,0030,0047,013D,C ;33014       M[TEMP.R]_MB.CLR1B              ;Set up to have WORD-WRITE code do
                                ;33015                                          ;remainder of WRITE for us
                                ;33016
                                ;33017          ;-------------------------------;
                                ;33018          VA_M[VA]-ZLIT0[2],              ;Point VA to last WORD, split into
U 13DC, 0581,BC10,0970,14A7,012E,0 ;33019       PS[<C>?,NEXT/DS.WRITE.WRD       ;WORD-WRITE code
```

```
                        ;33020          ;*****************************************************************************
                        ;33021          ; Continuation, last LONGWORD
                        ;33022          ;*****************************************************************************
                        ;33023 =0
                        ;33024 DS.WRITE.LNG:
                        ;33025          ;0-------------------------------;
                        ;33026          WB_M[TEMP.R].AND.ZLIT0[0F0],    ;Even destination, is high nibble
U 12EC, 0D81,0C12,0A37,8047,012E,E      ;33027          WX.EQ.0?,NEXT/DS.WRITE.LNG.EV   ;nonzero?
                        ;33028
                        ;33029          ;1-------------------------------;
                        ;33030          WRITE M[TEMP.R],SIZE[LONG],     ;Odd dest., WRITE it,
U 12ED, 0481,0592,4420,05D8,012F,0      ;33031          FLAG0?,NEXT/DS.WRITE.LNG.EX     ;is this first WRITE?
                        ;33032
                        ;33033 =0
                        ;33034 DS.WRITE.LNG.EV:
                        ;33035          ;0-------------------------------;
                        ;33036          SET FLAG1,                      ;Even dest., high nibble nonzero,
U 12EE, 0D4F,0C13,8037,8047,012E,F      ;33037          M[TEMP.R]_MB.ANDNOT.ZLIT0[0F0]  ;<V>image_1, zero high nibble
                        ;33038
                        ;33039          ;1-------------------------------;
                        ;33040          WRITE M[TEMP.R],FLAG0?,         ;(high nib zero,) WRITE it, is this
U 12EF, 0481,0592,4420,05D8,012F,0      ;33041          SIZE[LONG]                      ;first WRITE?
                        ;33042
                        ;33043 =0
                        ;33044 DS.WRITE.LNG.EX:
                        ;33045          ;0-------------------------------;
                        ;33046          WB_M[TEMP.R],WX.NE.0?,          ;Last longword, is it nonzero?
U 12F0, 0881,0592,4A70,0047,012F,2      ;33047          NEXT/DS.WRITE.EXIT
                        ;33048
                        ;33049          ;1-------------------------------;
                        ;33050          CLEAR FLAG0,WX.NE.0?,           ;First longword, is it nonzero?
U 12F1, 0101,0C93,8A70,7847,012F,2      ;33051          WB_M[TEMP.R].ANDNOT.ZLIT24[0F]  ;(no sign nibble)
```

```
                                ;33052          ;*****************************************************************************
                                ;33053          ; Common DS.WRITE exit.  Last WRITE has been completed, here from
                                ;33054          ; WX.NE.0? BUT on written data.  =0 is last WRITE was zero, =1 is
                                ;33055          ; nonzero.
                                ;33056          ;*****************************************************************************
                                ;33057 =0
                                ;33058 DS.WRITE.EXIT:
                                ;33059          ;0-----------------------------;
                                ;33060          WB_D+ZLIT0[1],WB<5-0>?,         ;What is the next TEMP number?
u 12F2, 0D80,0C31,0230,0847,0912,C 377*  ;33061          NEXT/DS.WRITE.EXIT.OV
                                ;33062
                                ;33063          ;1-----------------------------;
                                ;33064          CLEAR FLAG2,WB_D+ZLIT0[1],      ;<Z>image_0, what is next TEMP number?
u 12F3, 0D10,0C31,0230,0847,0912,C 377*  ;33065          WB<5-0>?
                                ;33066
                                ;33067          ;*****************************************************************************
                                ;33068          ; Check unwritten TEMPs for overflow.
                                ;33069          ;*****************************************************************************
                                ;33070 =100
                                ;33071 DS.WRITE.EXIT.OV:
                                ;33072          ;100---------------------------;
u 112C, 0486,5002,4031,0047,0112,D       ;33073          M[TEMP5]_MB.OR.R[TEMP4]         ;Are TEMP4-TEMP6 zero?
                                ;33074
                                ;33075          ;101---------------------------;
                                ;33076          CLEAR MM.NOINT,                 ;WAS SET AT START OF WRITE ROUTINE
                                ;33077          WB_M[TEMP5].OR.R[TEMP6],        ;(Are TEMP5 & TEMP6 zero?)
u 112D, 0C20,5002,4A71,8047,012F,4       ;33078          WX.NE.0?,NEXT/DS.WRITE.EXIT.OV.6
                                ;33079
                                ;33080          ;110---------------------------;
                                ;33081          CLEAR MM.NOINT,         ;
                                ;33082          WB_M[TEMP6],WX.NE.0?,           ;TEMP6?
u 112E, 0C20,6592,4A70,0047,012F,4       ;33083          NEXT/DS.WRITE.EXIT.OV.6
                                ;33084
                                ;33085          ;111---------------------------;
                                ;33086          CLEAR MM.NOINT,         ;
u 112F, 0020,0036,40B0,0047,03FF,F       ;33087          RETURN [-1]                     ;We wrote TEMP3-5 and some part of
                                ;33088                                          ;TEMP6, RETURN -1
                                ;33089
                                ;33090 =0
                                ;33091 DS.WRITE.EXIT.OV.6:
                                ;33092          ;0-----------------------------;
u 12F4, 0080,0036,40B0,0047,03FF,F       ;33093          RETURN [-1]                     ;No Ov, return
                                ;33094
                                ;33095          ;1-----------------------------;
u 12F5, 0848,0036,40B0,0047,03FF,F       ;33096          SET FLAG1,RETURN [-1]           ;Some TEMP(s) is >0, Ov., return
```

```
                                ;33097 .TOC "  Decimal String                    : Small Routines"
                                ;33098
                                ;33099          ;*****************************************************************************
                                ;33100          ; Small Routines are used at various locations in the Decimal String
                                ;33101          ; microcode and are for the most part visually apparent.  Note that
                                ;33102          ; some routines are part of other routines.
                                ;33103          ;*****************************************************************************
                                ;33104
                                ;33105 .REGION/DECMAL.R1L,DECMAL.R1H/DECMAL.R2L,DECMAL.R2H
                                ;33106
                                ;33107
                                ;33108          ;*****************************************************************************
                                ;33109          ; DS.SR.BIN.MUL10. adds the binary value in QREG to the binary value
                                ;33110          ; in TEMP7 and multiplies it by 10.  DS.SR.BIN.MUL10..2 is the trail
                                ;33111          ; end.  Both are used in MULP.
                                ;33112          ;*****************************************************************************
                                ;33113 DS.SR.BIN.MUL10.:
                                ;33114          ;--------------------------------;
U 13DD, 0884,0039,C031,C047,013D,E  ;33115          R[TEMP7]_(RB+Q).SL.1           ;TEMP7_<data>'0
                                ;33116
                                ;33117 DS.SR.BIN.MUL10..2:
                                ;33118          ;--------------------------------;
                                ;33119          M[TEMP7]_MB+(R[TEMP7].SL.2),    ;TEMP7_<data>0
U 13DE, 0886,75D1,00A1,C047,0000,1  ;33120          RETURN [1]
                                ;33121
                                ;33122
                                ;33123          ;*****************************************************************************
                                ;33124          ; Used by DIVP to zero appropriate bytes in GPR0, GPR2, and GPR4.
                                ;33125          ;*****************************************************************************
                                ;33126 DS.SR.26:
                                ;33127          ;--------------------------------;
U 13DF, 0186,0C37,0037,F847,013E,0  ;33128          M[TEMP0]_ZLIT0[0FF]            ;Byte mask
                                ;33129
                                ;33130          ;--------------------------------;
U 13E0, 0884,03BE,0004,0047,013E,1  ;33131          R[R0]_RB.AND.(M[TEMP0].RR.8)   ;GPR0_zero'd low 3 bytes
                                ;33132
                                ;33133          ;--------------------------------;
U 13E1, 0884,03BE,0015,0047,013E,2  ;33134          R[R4]_RB.AND.(M[TEMP0].RR.16)  ;Zero out bytes #0, 1, and 3 of GPR4
                                ;33135
                                ;33136          ;--------------------------------;
                                ;33137          R[R2]_M[TEMP0].AND.RB,          ;Zero out high 3 bytes, GPR2,
U 13E2, 0C84,0002,00B4,8047,0000,1  ;33138          RETURN [1]                     ;RETURN +1
```

```
                                ;33139          ;*************************************************************************
                                ;33140          ; DS.SR.18 is used by DIVP code to manipulate pieces of fractional
                                ;33141          ; LONGWORDs.  RETURNs +2 if BUS exception occurs.
                                ;33142          ;*************************************************************************
                                ;33143 =0**
                                ;33144 DS.SR.18:
                                ;33145          ;0**-------------------------------;
                                ;33146          R[TEMP13]_ZEXT(XB) PC_PC+1,     ;TEMP13_next BYTE
U 11BA, 0885,759E,4003,6047,013E,3  ;33147      NEXT/DS.SR.18.CON
                                ;33148
                                ;33149          ;1**-------------------------------;
U 11BE, 0586,3C37,00B0,1047,0000,2  ;33150      M[TEMP3]_ZLIT0[2],RETURN [2]    ;RETURN +4 from BUS exception, RETURN +2
                                ;33151                                          ;from routine, int. code #2
                                ;33152
                                ;33153 DS.SR.18.CON:
                                ;33154          ;-------------------------------;
                                ;33155          M[TEMP.R]_MB.OR.(R[TEMP13].RR.8),       ;Mask in new byte into byte#3
U 13E3, 0887,0292,4083,4047,0000,1  ;33156      RETURN [1]                      ;of current TEMP
                                ;33157
                                ;33158
                                ;33159          ;*************************************************************************
                                ;33160          ; Used by DIVP to do packing and StateSave operations on GPR2.
                                ;33161          ;*************************************************************************
                                ;33162 DS.SR.21:
                                ;33163          ;-------------------------------;
U 13E4, 0484,13BE,4004,8047,013E,5  ;33164      R[R2]_RB.OR.(M[TEMP1].RR.8)     ;GPR2, byte#3_TEMP1
                                ;33165
                                ;33166          ;-------------------------------;
U 13E5, 0484,53BE,4024,8047,013E,6  ;33167      R[R2]_RB.OR.(M[TEMP5].RR.24)    ;GPR2, byte#1_TEMP5
                                ;33168
                                ;33169 DS.SR.28:
                                ;33170          ;-------------------------------;
                                ;33171          R[TEMP3]_ZEXT(M[TEMP4]),        ;TEMP4 could be negative (DIVP)
U 13E6, 0884,459E,4000,C047,013E,7  ;33172      SIZE[BYTE]
                                ;33173
                                ;33174          ;-------------------------------;
                                ;33175          R[R2]_RB.OR.(M[TEMP3].RR.16),   ;GPR2, byte#2_ZEXT(TEMP4)
U 13E7, 0884,33BE,4094,8047,0000,1  ;33176      RETURN [1]
```

```
                                ;33177          ;*****************************************************************************
                                ;33178          ; BCDSWaP MTEMP8
                                ;33179          ;*****************************************************************************
                                ;33180 DS.SR.22:
                                ;33181          ;-------------------------------;
U 13E8, 0C86,8637,00B0,0047,0000,1      ;33182          M[TEMP8]_MB.BCDSWP,RETURN [1]
                                ;33183
                                ;33184
                                ;33185          ;*****************************************************************************
                                ;33186          ; BCDSWaP MTEMP9
                                ;33187          ;*****************************************************************************
                                ;33188 DS.SR.23:
                                ;33189          ;-------------------------------;
U 13E9, 0886,9637,00B0,0047,0000,1      ;33190          M[TEMP9]_MB.BCDSWP,RETURN [1]
                                ;33191
                                ;33192
                                ;33193          ;*****************************************************************************
                                ;33194          ; BCDSWaP MTEMP10
                                ;33195          ;*****************************************************************************
                                ;33196 DS.SR.24:
                                ;33197          ;-------------------------------;
U 13EA, 0886,A637,00B0,0047,0000,1      ;33198          M[TEMP10]_MB.BCDSWP,RETURN [1]  ;BCDSWP MTEMP10
                                ;33199
                                ;33200
                                ;33201
                                ;33202          ;*****************************************************************************
                                ;33203          ; Used in DIVP to shift a LONGWORD by a digit (x 10.)
                                ;33204          ;*****************************************************************************
                                ;33205 DS.SR.25:
                                ;33206          ;-------------------------------;
                                ;33207          Q_(R[TEMP7] M[TEMP3]).RL.4,     ;Q_''new'' TEMP3
U 13EB, 0C80,3237,10B1,C047,0000,1      ;33208          RETURN [1]
```

```
                                        ;33209          ;*****************************************************************************
                                        ;33210          ; Used in DS.DIV10 and DS.MUL10 to BCDSWaP data to and from arithmetic
                                        ;33211          ; format.
                                        ;33212          ;*****************************************************************************
                                        ;33213 DS.SR.456SWP:
                                        ;33214          ;-------------------------------;
U 13EC, 0886,4637,0030,0047,013E,D      ;33215          M[TEMP4]_MB.BCDSWP
                                        ;33216
                                        ;33217          ;-------------------------------;
U 13ED, 0C86,5637,0030,0047,013E,E      ;33218          M[TEMP5]_MB.BCDSWP
                                        ;33219
                                        ;33220          ;-------------------------------;
U 13EE, 0886,6637,00B0,0047,0000,1      ;33221          M[TEMP6]_MB.BCDSWP,RETURN [1]
                                        ;33222
                                        ;33223
                                        ;33224          ;*****************************************************************************
                                        ;33225          ; Used in DS.DIV10.DIVP to BCDSWaP data to & from arithmetic format.
                                        ;33226          ;*****************************************************************************
                                        ;33227 DS.SR.123SWP:
                                        ;33228          ;-------------------------------;
U 13EF, 0886,1637,0030,0047,013F,0      ;33229          M[TEMP1]_MB.BCDSWP
                                        ;33230
                                        ;33231          ;-------------------------------;
U 13F0, 0086,2637,0030,0047,013F,1      ;33232          M[TEMP2]_MB.BCDSWP
                                        ;33233
                                        ;33234          ;-------------------------------;
U 13F1, 0886,3637,00B0,0047,0000,1      ;33235          M[TEMP3]_MB.BCDSWP,RETURN [1]
                                        ;33236
                                        ;33237
                                        ;33238          ;*****************************************************************************
                                        ;33239          ; Used by DIVP to mask in FLAGS.
                                        ;33240          ;*****************************************************************************
                                        ;33241
                                        ;33242 DS.SR.GPR0IN.FLAGS:
                                        ;33243          ;-------------------------------;
                                        ;33244          R[R0].SIZ_FLAGS,RETURN [1],     ;FLAGS, GPR0 is done
U 13F2, 0882,0036,4084,0387,0000,1      ;33245          SIZE[BYTE]
```

```
                                        ;33246 .TOC   ""      Decimal String                    : Native Mode IRD Specs"
                                        ;33247
;33248  .NOBIN
;33249
;33250  .ICODE  ;       OPS             REG                     MEM             OPS         FPA REG                 FPA MEM
;33251        20: FPD [NOP][DS.DS.UNP                 ]                       [NOP][DS.DS.UNP                 ],          ;ADDP4
;33252            IRD1[LOD][OS.RED                    ]                       [LOD][OS.RED                    ]
;33253  .OCODE
;33254        20: CNT0[NOP][DS.CMPP4                  ][DS.CMPP4              ] [NOP][DS.CMPP4                  ][DS.CMPP4              ],
;33255            CNT1[NOP][DS.CMPP4.S2               ][DS.CMPP4.S2           ] [NOP][DS.CMPP4.S2               ][DS.CMPP4.S2           ]
;33256
;33257  .ICODE
;33258        21: FPD [NOP][DS.DS.UNP                 ]                       [NOP][DS.DS.UNP                 ],          ;ADDP6
;33259            IRD1[LOD][OS.RED                    ]                       [LOD][OS.RED                    ]
;33260  .OCODE
;33261        21: CNT0[NOP][DS.ADDP6                  ][DS.ADDP6              ] [NOP][DS.ADDP6                  ][DS.ADDP6              ],
;33262            CNT1[NOP][DS.ADDP6.2                ][DS.ADDP6.2            ] [NOP][DS.ADDP6.2                ][DS.ADDP6.2            ]
;33263
;33264  .ICODE
;33265       0F8: FPD [NOP][DS.DS.UNP                 ]                       [NOP][DS.DS.UNP                 ],          ;ASHP
;33266            IRD1[LOD][OS.RED                    ]                       [LOD][OS.RED                    ]
;33267  .OCODE
;33268       0F8: CNT0[NOP][DS.ASHP                   ][DS.ASHP               ] [NOP][DS.ASHP                   ][DS.ASHP               ],
;33269            CNT1[NOP][DS.ASHP.2                 ][DS.ASHP.2             ] [NOP][DS.ASHP.2                 ][DS.ASHP.2             ]
;33270
;33271  .ICODE
;33272       035: FPD [NOP][DS.DS.UNP                 ]                       [NOP][DS.DS.UNP                 ],          ;CMPP3
;33273            IRD1[LOD][OS.RED                    ]                       [LOD][OS.RED                    ]
;33274  .OCODE
;33275       035: CNT0[NOP][DS.MOVP                   ][DS.MOVP               ] [NOP][DS.MOVP                   ][DS.MOVP               ],
;33276            CNT1[NOP][DS.MOVP.2                 ][DS.MOVP.2             ] [NOP][DS.MOVP.2                 ][DS.MOVP.2             ]
;33277
;33278  .ICODE
;33279       037: FPD [NOP][DS.DS.UNP                 ]                       [NOP][DS.DS.UNP                 ],          ;CMPP4
;33280            IRD1[LOD][OS.RED                    ]                       [LOD][OS.RED                    ]
;33281  .OCODE
;33282       037: CNT0[NOP][DS.CMPP4                  ][DS.CMPP4              ] [NOP][DS.CMPP4                  ][DS.CMPP4              ],
;33283            CNT1[NOP][DS.CMPP4.S2               ][DS.CMPP4.S2           ] [NOP][DS.CMPP4.S2               ][DS.CMPP4.S2           ]
;33284
;33285  .ICODE
;33286       0F9: FPD [NOP][IE.OPCOD.DEC              ]                       [NOP][IE.OPCOD.DEC              ],          ;CVTLP
;33287            IRD1[LOD][OS.RED                    ]                       [LOD][OS.RED                    ]
;33288  .OCODE
;33289       0F9: CNT0[LOD][OS.RED                    ][OS.RED                ] [LOD][OS.RED                    ][OS.RED                ],
;33290            CNT1[NOP][DS.CVTLP                  ][DS.CVTLP              ] [NOP][DS.CVTLP                  ][DS.CVTLP              ]
;33291
;33292  .ICODE
;33293       036: FPD [NOP][DS.CVTPL.UNPACK           ]                       [NOP][DS.CVTPL.UNPACK           ],          ;CVTPL
;33294            IRD1[LOD][OS.RED                    ]                       [LOD][OS.RED                    ]
;33295  .OCODE
;33296       036: CNT0[NOP][DS.CVTPL                  ][DS.CVTPL              ] [NOP][DS.CVTPL                  ][DS.CVTPL              ],
;33297            CNT1[NOP][DS.CVTPL.02               ][DS.CVTPL.02           ] [NOP][DS.CVTPL.02               ][DS.CVTPL.02           ]
```

```
;33298  .ICODE  ;      OPS          REG                    MEM                    OPS        FPA REG                FPA MEM
;33299       026: FPD [NOP][DS.CVTXX.UNPA K ]                                  [NOP][DS.CVTXX.UNPACK ],                          ;CVTTP
;33300            IRD1[LOD][OS.RED           ]                                  [LOD][OS.RED          ]
;33301  .OCODE
;33302       026: CNT0[NOP][DS.CVTTP         ][DS.CVTTP         ]  [NOP][DS.CVTTP         ][DS.CVTTP         ],
;33303            CNT1[NOP][DS.CVTTP.02      ][DS.CVTTP.02      ]  [NOP][DS.CVTTP.02      ][DS.CVTTP.02      ]
;33304
;33305  .ICODE
;33306       024: FPD [NOP][DS.CVTXX.UNPACK ]                                  [NOP][DS.CVTXX.UNPACK ],                          ;CVTPT
;33307            IRD1[LOD][OS.RED          ]                                  [LOD][OS.RED          ]
;33308  .OCODE
;33309       024: CNT0[NOP][DS.CVTPT         ][DS.CVTPT         ]  [NOP][DS.CVTPT         ][DS.CVTPT         ],
;33310            CNT1[NOP][DS.CVTPT.02      ][DS.CVTPT.02      ]  [NOP][DS.CVTPT.02      ][DS.CVTPT.02      ]
;33311
;33312  .ICODE
;33313       008: FPD [NOP][DS.CVTXX.UNPACK ]                                  [NOP][DS.CVTXX.UNPACK ],                          ;CVTPS
;33314            IRD1[LOD][OS.RED          ]                                  [LOD][OS.RED          ]
;33315  .OCODE
;33316       008: CNT0[NOP][DS.CVTPS         ][DS.CVTPS         ]  [NOP][DS.CVTPS         ][DS.CVTPS         ],
;33317            CNT1[NOP][DS.CVTPS.02      ][DS.CVTPS.02      ]  [NOP][DS.CVTPS.02      ][DS.CVTPS.02      ]
;33318
;33319  .ICODE
;33320       009: FPD [NOP][DS.CVTXX.UNPACK ]                                  [NOP][DS.CVTXX.UNPACK ],                          ;CVTSP
;33321            IRD1[LOD][OS.RED          ]                                  [LOD][OS.RED          ]
;33322  .OCODE
;33323       009: CNT0[NOP][DS.CVTSP         ][DS.CVTSP         ]  [NOP][DS.CVTSP         ][DS.CVTSP         ],
;33324            CNT1[NOP][DS.CVTSP.02      ][DS.CVTSP.02      ]  [NOP][DS.CVTSP.02      ][DS.CVTSP.02      ]
;33325
;33326  .ICODE
;33327       027: FPD [NOP][DS.DIVP.UNP      ]                                  [NOP][DS.DIVP.UNP     ],                          ;DIVP
;33328            IRD1[LOD][OS.RED           ]                                  [LOD][OS.RED          ]
;33329  .OCODE
;33330       027: CNT0[NOP][DS.ADDP6         ][DS.ADDP6         ]  [NOP][DS.ADDP6         ][DS.ADDP6         ],
;33331            CNT1[NOP][DS.ADDP6.2       ][DS.ADDP6.2       ]  [NOP][DS.ADDP6.2       ][DS.ADDP6.2       ]
;33332
;33333  .ICODE
;33334       034: FPD [NOP][DS.DS.UNP        ]                                  [NOP][DS.DS.UNP       ],                          ;MOVP
;33335            IRD1[LOD][OS.RED           ]                                  [LOD][OS.RED          ]
;33336  .OCODE
;33337       034: CNT0[NOP][DS.MOVP          ][DS.MOVP          ]  [NOP][DS.MOVP          ][DS.MOVP          ],
;33338            CNT1[NOP][DS.MOVP.2        ][DS.MOVP.2        ]  [NOP][DS.MOVP.2        ][DS.MOVP.2        ]
;33339
;33340  .ICODE
;33341       025: FPD [NOP][DS.DS.UNP        ]                                  [NOP][DS.DS.UNP       ],                          ;MULP
;33342            IRD1[LOD][OS.RED           ]                                  [LOD][OS.RED          ]
;33343  .OCODE
;33344       025: CNT0[NOP][DS.ADDP6         ][DS.ADDP6         ]  [NOP][DS.ADDP6         ][DS.ADDP6         ],
;33345            CNT1[NOP][DS.ADDP6.2       ][DS.ADDP6.2       ]  [NOP][DS.ADDP6.2       ][DS.ADDP6.2       ]
```

```
;33346  .ICODE  ;      OPS          REG                    MEM                OPS          FPA REG                FPA MEM
;33347       022: FPD [NOP][DS.DS.UNP          ]                          [NOP][DS.DS.UNP          ],                    ;SUBP4
;33348            IRD1[LOD][OS.RED             ]                          [LOD][OS.RED             ]
;33349  .OCODE
;33350       022: CNT0[NOP][DS.CMPP4           ][DS.CMPP4           ]  [NOP][DS.CMPP4           ][DS.CMPP4           ],
;33351            CNT1[NOP][DS.CMPP4.S2        ][DS.CMPP4.S2        ]  [NOP][DS.CMPP4.S2        ][DS.CMPP4.S2        ]
;33352
;33353  .ICODE
;33354       023: FPD [NOP][DS.DS.UNP          ]                          [NOP][DS.DS.UNP          ],                    ;SUBP6
;33355            IRD1[LOD][OS.RED             ]                          [LOD][OS.RED             ]
;33356  .OCODE
;33357       023: CNT0[NOP][DS.ADDP6           ][DS.ADDP6           ]  [NOP][DS.ADDP6           ][DS.ADDP6           ],
;33358            CNT1[NOP][DS.ADDP6.2         ][DS.ADDP6.2         ]  [NOP][DS.ADDP6.2         ][DS.ADDP6.2         ]
```

```
;33359  .TOC     "      Decimal String                  : DSIZE ROM specs"
;33360
;33361  .DCODE
;33362
;33363  020:    SIZE [WORD] [BYTE] [WORD] [BYTE] [   0] [   0]  ;ADDP4
;33364
;33365  021:    SIZE [WORD] [BYTE] [WORD] [BYTE] [WORD] [BYTE]  ;ADDP6
;33366
;33367  0F8:    SIZE [BYTE] [WORD] [BYTE] [BYTE] [WORD] [BYTE]  ;ASHP
;33368
;33369  035:    SIZE [WORD] [BYTE] [BYTE] [   0] [   0] [   0]  ;CMPP3
;33370
;33371  037:    SIZE [WORD] [BYTE] [WORD] [BYTE] [   0] [   0]  ;CMPP4
;33372
;33373  0F9:    SIZE [LONG] [WORD] [BYTE] [   0] [   0] [   0]  ;CVTLP
;33374
;33375  036:    SIZE [WORD] [BYTE] [LONG] [   0] [   0] [   0]  ;CVTPL
;33376
;33377  026:    SIZE [WORD] [BYTE] [BYTE] [WORD] [BYTE] [   0]  ;CVTTP
;33378
;33379  024:    SIZE [WORD] [BYTE] [BYTE] [WORD] [BYTE] [   0]  ;CVTPT
;33380
;33381  008:    SIZE [WORD] [BYTE] [WORD] [BYTE] [   0] [   0]  ;CVTPS
;33382
;33383  009:    SIZE [WORD] [BYTE] [WORD] [BYTE] [   0] [   0]  ;CVTSP
;33384
;33385  027:    SIZE [WORD] [BYTE] [WORD] [BYTE] [WORD] [BYTE]  ;DIVP
;33386
;33387  034:    SIZE [WORD] [BYTE] [BYTE] [   0] [   0] [   0]  ;MOVP
;33388
;33389  025:    SIZE [WORD] [BYTE] [WORD] [BYTE] [WORD] [BYTE]  ;MULP
;33390
;33391  022:    SIZE [WORD] [BYTE] [WORD] [BYTE] [   0] [   0]  ;SUBP4
;33392
;33393  023:    SIZE [WORD] [BYTE] [WORD] [BYTE] [WORD] [BYTE]  ;SUBP6
;33394
;33395
;33396
;33397          ;*****************************************************************
;33398          ;
;33399          ; End of DECMAL
;33400          ;
;33401          ;*****************************************************************
;33402
;33403  .DCODE
                                        ;33404 .BIN
```

```
                                        ;33405 .TOC "EDIT.MIC"
                                        ;33406 .TOC "REVISION 13.0"
                                        ;33407 ;       Gerard Koeckhoven, CHARLIE MCDOWELL
                                        ;33408
;33409  .NOBIN
;33410
;33411  .TOC "  Revision History"
;33412
;33413  ; REV   EXPLAINATION
;33414
;33415  ; 13    Fix EDITPC to get proper data in R5
;33416  ; 12    Fix that R0 and R2 don't get clobbered.
;3341   ; 11    Fix REPLACE_SIGN to work if a page fault occurs doing the fix up.
;33418  ; 10    Initial release.
                                        ;33419         .BIN
```

```
;33420 .TOC "  Edit Packed to Character string : INITIALIZE"
;33421
;33422 ;*****************************************************************************
;33423 ;       EDITPC  srclen.rw, srcaddr.ab, pattern.ab, dstaddr.ab
;33424 ;
;33425 ;       Input           MDR             Source length (srclen).
;33426 ;
;33427 ;       Resources       MDR             Fetch other operands
;33428 ;                       Q               Save operands
;33429 ;                       VA
;33430 ;                       R3              Pattern address
;33431 ;                       TEMP2           Temporary
;33432 ;                       TEMP3           Temporary
;33433 ;
;33434 ;       Output          TEMP0           Pattern byte from XB
;33435 ;                       TEMP1           Save source length
;33436 ;                       TEMP2           Save PC
;33437 ;                       TEMP3           Return code for FPD
;33438 ;                       TEMP4           Save sign byte
;33439 ;                       MTEMP8          Save PSL for easy access of CC's
;33440 ;                       MTEMP9          Save ASCII '0' for conversion
;33441 ;                       FPDOFFSET       Set to 0 for all EDITPC
;33442 ;                       PC              Used to prefetch pattern bytes
;33443 ;                       R0              Source length remaining
;33444 ;                       R2              Sign'Fill register
;33445 ;                       R5              Next destination byte address
;33446 ;
;33447 ;       Subroutines     IE.OPER.FAULT
;33448 ;                       OS.ADD
;33449 ;                       ED.POD.10
;33450 ;                       IE.SERV.IP.TS
;33451 ;                       ED.PACK
;33452 ;
;33453 ;       This section contains the initialization code for EDITPC and also
;33454 ;       contains the main "pattern operator" loop (ED.LOOP).
;33455 ;       Following this is a section to do the "pattern operator decode"
;33456 ;       (ED.POD), seperate sections for each pattern operator and read
;33457 ;       and store routines very similiar to those described in the SRM.
;33458 ;
;33459 ;       CHARLIE'S FLOWS 6.1
;33460 ;*****************************************************************************
;33461
;33462 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
;33463 ED.EDITPC:
;33464                   ;---------------------------------;
;33465                   WB_SEXT(M[MDR]).ANDNOT.ZLITO[1F],       ; CHECK IF SRCLEN GRTU 31
;33466                   SIZE[WORD],                     ;
U 03C1, 0581,2C1F,8A10,F847,014E,8  ;33467              WX.EQ.0?                        ;
;33468 .REGION/EDIT.R1L,EDIT.R1H/EDIT.R2L,EDIT.R2H/EDIT.R3L,EDIT.R3H
;33469
;33470 :0          ;0---------------------------------;
U 14E8, 0C80,0036,4030,0047,00FF,8  ;33471              NEXT/IE.OPER.FAULT              ; SRCLEN IS GRTU THEN 31
```

```
                                ;33472          ;1-------------------------------;
                                ;33473          R[TEMP1]_ZEXT(M[MDR]),          ;
                                ;33474          SIZE[WORD],                     ; SAVE SRCLEN
U 14E9, 0485,259E,4190,4047,0012,0      ;33475          LOD INC BRA?,NEXT/OS.ADD        ; FETCH SRCADDR (RETURN VIA IRDX ROM).
                                ;33476
                                ;33477 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;33478
                                ;33479 =00
                                ;33480 ED.EDITPC.02:
                                ;33481          ;00-------------------------------;
                                ;33482          M[FPDOFFSET]_ZLIT0[0],          ; SET FPD PACK BRANCH OFFSET
                                ;33483          LOD INC BRA?,                   ;
U 02BC, 0586,CC37,01B0,0047,0412,0      ;33484          PUSH,NEXT/OS.ADD                ; FETCH PATTERN ADDRESS
                                ;33485
                                ;33486          ;01-------------------------------;
                                ;33487          M[TEMP2]_Q,                     ; SAVE SRCADDR
                                ;33488          LOD INC BRA?,                   ;
U 02BD, 0886,203A,41BD,8047,0412,0      ;33489          PUSH,NEXT/OS.ADD                ; FETCH DESTINATION ADDRESS
                                ;33490
                                ;33491          ;10-------------------------------;
U 02BE, 0086,35BD,8030,4047,014E,4      ;33492          M[TEMP3]_R[TEMP1].SR.1          ; GET LENGTH OF SOURCE IN BYTES
                                ;33493 =
                                ;33494 .REGION/EDIT.R1L,EDIT.R1H/EDIT.R2L,EDIT.R2H/EDIT.R3L,EDIT.R3H
                                ;33495          ;-------------------------------;
U 14E4, 0080,3001,0030,84A7,014F,0      ;33496          VA_M[TEMP3]+R[TEMP2]            ; GET ADDRESS OF SOURCE SIGN BYTE.
                                ;33497
                                ;33498          ;-------------------------------;
                                ;33499          READ,SIZE[BYTE],                ; FETCH SOURCE SIGN BYTE.
U 14F0, 0C85,2592,4000,C050,0150,5      ;33500          R[TEMP3]_M[MDR]                 ; SAVE DESTINATION ADDRESS
                                ;33501
                                ;33502          ;-------------------------------;
U 1505, 0085,23B7,0001,0047,0150,8      ;33503          R[TEMP4]_M[MDR].RL.24           ; SAVE SIGN BYTE ROTATED FOR SIGN CHECK.
                                ;33504
                                ;33505          ;------ -------------------------;
U 1508, 0480,2002,403D,84A7,0151,0      ;33506          VA_M[TEMP2]                     ; LOAD SOURCE ADDRESS
                                ;33507
                                ;33508          ;-------------------------------;
                                ;33509          WB_R[TEMP1],                    ; INITIALIZE THE READ ROUTINE
U 1510, 0880,05BE,4270,4047,014E,A      ;33510          WB<1-0>?                        ; BRANCH ON RIGHT OR LEFT NIBBLE
                                ;33511 =10
                                ;33512          ;10-------------------------------;
                                ;33513          READ,SIZE[BYTE],                ;
                                ;33514          R[R1]_M[TEMP2],                 ;
U 14EA, 0884,2592,4004,4050,0151,8      ;33515          NEXT/ED.EDITPC.10               ;
                                ;33516
                                ;33517          ;11-------------------------------;
U 14EB, 0884,2E51,0034,4047,0151,8      ;33518          R[R1]_M[TEMP2]-1                ;
                                ;33519
                                ;33520 ED.EDITPC.10:
                                ;33521          ;-------------------------------;
U 1518, 0C84,1592,4034,0047,0152,E      ;33522          R[R0]_M[TEMP1]                  ; SAVE SOURCE LENGTH
                                ;33523
                                ;33524          ;-------------------------------;
U 152E, 0084,3592,4035,4047,0154,A      ;33525          R[R5]_M[TEMP3]                  ; SAVE DESTINATION ADDRESS
```

```
                        ;33526          ;--------------------------------;
U 154A, 0586,2C37,0031,0047,0155,6      ;33527          M[TEMP2]_ZLIT0[20]              ; INITIALIZE FILL CHARACTER TO BLANK.
                        ;33528
                        ;33529          ;--------------------------------; BRANCH ON SOURCE SIGN BYTE
U 1556, 0450,4636,4DF0,0047,0150,6      ;33530          BCD SIGN M[TEMP4]?,SET FLAG2    ; SET PSL<Z> FLAG.
                        ;33531
                        ;33532 =10      ;10-------------------------------;
                        ;33533          CC_M[TEMP3]_ZLIT0[0C],          ; CLEAR TEMP3 (EXCEPT LOW BYTE)
                        ;33534          SET FLAG3,                      ; SOURCE HAS MINUS SET N, Z, CLEAR V,C
U 1506, 015E,3C37,0030,60A7,0156,6      ;33535          NEXT/ED.MINUS           ;
                        ;33536
                        ;33537          ;11-------------------------------; CLEAR TEMP3 (EXCEPT LOW BYTE)
U 1507, 0D86,3C37,0030,20A7,0155,8      ;33538          CC_M[TEMP3]_ZLIT0[4]            ; SOURCE HAS PLUS SET N, CLEAR Z,V,C
                        ;33539
                        ;33540          ;--------------------------------;
                        ;33541          M[TEMP2]_MB.OR.ZLIT8[20],       ; INITIALIZE SIGN TO BLANK
U 1558, 0D86,2D92,4031,0047,0157,3      ;33542          NEXT/ED.SIGN.FILL               ;
                        ;33543 ED.MINUS:
                        ;33544          ;--------------------------------;
U 1566, 0586,2D92,4031,6847,0157,3      ;33545          M[TEMP2]_MB.OR.ZLIT8[2D]        ; INITIALIZE SIGN'FILL TO MINUS'BLANK
                        ;33546
                        ;33547 ED.SIGN.FILL:
                        ;33548          ;--------------------------------;
U 1573, 0084,2592,4034,8047,0157,A      ;33549          R[R2]_M[TEMP2]                  ;
                        ;33550
                        ;33551 ED.DONE.OS.EVAL:
                        ;33552          ;--------------------------------; OPERAND SPECIFIER EVALUATION IS
U 157A, 04ED,A592,4030,8047,0157,E      ;33553          R[TEMP2]_M[PC],SET FPD          ; COMPLETE, SAVE PC AND SET FPD
                        ;33554
                        ;33555          ;--------------------------------;
U 157E, 0084,002C,7034,C487,0158,B      ;33556          PC_R[R3]_Q Q_D                  ; USE XB TO PRE-FETCH PATTERN BYTES.
                        ;33557
                        ;33558          ;1--------------------------------;
U 1588, 0086,8036,4030,0087,014E,2      ;33559          M[TEMP8]_PSL,NEXT/ED.LOOP       ; KEEP PSL WHERE IT IS EASY TO GET AT.
                        ;33560 =000
                        ;33561 ED.POD:  ;000------------------------------;
                        ;33562          WB_(M[TEMP0] R[TEMP0]),RR,4,    ; BRANCH ON PATTERN BYTE<7:4>
U 14E0, 0880,0277,0230,0047,014F,8      ;33563          WB<5-0>?,NEXT/ED.POD.10         ;
                        ;33564
                        ;33565          ;001------------------------------; GO SERVICE THE INTERRUPT AND IF
                        ;33566          INTPEND OR TIMER?,              ;
U 14E1, 0480,0036,4AF0,0047,04F7,0      ;33567          PUSH,NEXT/IE.SERV.IP.TS2        ; IT WAS THE TIMER, RETURN - 1
                        ;33568
                        ;33569 ED.LOOP:
                        ;33570          ;010------------------------------;
                        ;33571          R[TEMP0]_ZEXT(XB) PC_PC+1,      ; FETCH PATTERN BYTE
U 14E2, 0085,759E,4B00,38E7,014E,0      ;33572          IP.TS?,NEXT/ED.POD              ; IF NO INTERRUPTS, DECODE THE PATTERN
                        ;33573
                        ;33574 =101     ;101------------------------------; FPD PACK RETURN FROM INTERRUPT CODE,
U 14E5, 0D81,AC10,0030,0C87,014E,6      ;33575          PC_M[PC]-ZLIT0[1]               ; THROW AWAY LAST PATTERN BYTE FETCH
                        ;33576
                        ;33577          ;110------------------------------; FPD PACK RETURN FROM UTRAPPED
                        ;33578          M[TEMP3]_ZLIT8[0A],             ; SOURCE XB AT ED.LOOP, SET FPD CODE,
U 14E6, 0586,3D87,0030,5047,0165,0      ;33579          NEXT/ED.PACK                    ;D GO PACK IT UP.
                        ;33580 =
```

```
                        ;33581 .TOC "  Edit Packed to Character string : PATTERN OPERATOR DECODE"
                        ;33582
                        ;33583 ;*****************************************************************************
                        ;33584 ;       Entry points    ED.POD.10
                        ;33585 ;
                        ;33586 ;       Input           TEMP0           Pattern operator
                        ;33587 ;
                        ;33588 ;       Output          TEMP3           Count for EO$FILL, EO$MOVE and EO$FLOAT
                        ;33589 ;                                       or return code if reserved operand.
                        ;33590 ;                       TEMP6           Byte operand for operators 40-47
                        ;33591 ;
                        ;33592 ;       Subroutines     ED.RESV.OPER
                        ;33593 ;                       ED.40-47
                        ;33594 ;                       ED.PACK
                        ;33595 ;                       ED.00-04
                        ;33596 ;                       ED.FILL
                        ;33597 ;                       ED.MOVE
                        ;33598 ;                       ED.FLOAT
                        ;33599 ;
                        ;33600 ;       This routine decodes the pattern operator and passes control
                        ;33601 ;       to the proper routine to process that operator.
                        ;33602 ;
                        ;33603 ;       Charlie's flows 6.2
                        ;33604 ;*****************************************************************************
                        ;33605
                        ;33606 =111011
                        ;33607 ED.POD.10:
                        ;33608          ;111011-------------------------; PO = 0X, 1X, 2X, 3X, 8X, 9X, AX, BX
                        ;33609          WB_M[TEMP0]-ZLIT0[5],           ; ISOLATE 00-04
                        ;33610          SIGND CMP DEF?,SIZE[LONG],      ;
U 14FB, 0580,0C10,0860,2847,0950,9 413* ;33611          NEXT/ED.POD.20          ;
                        ;33612
                        ;33613          ;111111-------------------------; PO = 4X, 5X, 6X, 7X, CX, DX, EX, FX
                        ;33614          WB_M[TEMP0]-ZLIT0[48],          ; ISOLATE 40-47
U 14FF, 0980,0C10,0862,4047,094F,1 413* ;33615          SIGND CMP DEF?,SIZE[LONG]       ;
                        ;33616
                        ;33617 =001     ;001----------------------------; PO = 48-4F, 5X, 6X, 7X, CX, DX, EX, FX
                        ;33618          M[TEMP3]_ZLIT8[0A],             ;
U 14F1, 0D86,3DB7,0030,5047,0151,A      ;33619          NEXT/ED.RESV.OPER       ;
                        ;33620
                        ;33621 ED.POD.15:
                        ;33622          ;011----------------------------; PO = 40-47
                        ;33623          R[TEMP6]_ZEXT(XB) PC_PC+1,      ;
U 14F3, 0C85,759E,4001,A047,0159,8      ;33624          NEXT/ED.40-47           ;
                        ;33625
                        ;33626 =111
                        ;33627 ED.POD.16:
                        ;33628          ;111----------------------------; FPD PACK RETURN FROM UTRAPPED
U 14F7, 0586,3DB7,0030,4847,0158,E      ;33629          M[TEMP3]_ZLIT8[9]               ; SOURCE XB AT .-4, SET FPD CODE,
                        ;33630
                        ;33631          ;-------------------------------;
                        ;33632          M[TEMP3]_MB.OR.R[TEMP0],        ;
U 158E, 0886,3002,4030,0047,0165,0      ;33633          NEXT/ED.PACK            ; SAVE PATTERN OPERATOR, GO PACK IT UP.
```

```
                                ;33634 =01
                                ;33635 ED.POD.20:
                                ;33636          ;01------------------------------; PO = 05-0F, 1X, 2X, 3X, 8X, 9X, AX, BX
                                ;33637          WB_M[TEMP0]-ZLIT0[81],          ; ISOLATE FILL, MOVE AND FLOAT
                                ;33638          SIZE[LONG],                     ;
U 1509, 0180,0C10,0B64,0847,0951,9 413* ;33639  SIGND CMP DEF?,NEXT/ED.POD.30   ;
                                ;33640
                                ;33641          ;11------------------------------; PO = 00-04
                                ;33642          WB_R[TEMP0],                    ;
U 150B, 0880,05BE,4230,0047,0150,0      ;33643  WB<5-0>?,NEXT/ED.00-04          ; SEPERATE 00,01,02,03, AND 04
                                ;33644
                                ;33645 =01
                                ;33646 ED.POD.30:
                                ;33647          ;01------------------------------; PO = 81-8F, 9X, AX, BX
                                ;33648          WB_M[TEMP0]-ZLIT0[0B0],         ; ISOLATE FILL, MOVE AND FLOAT
                                ;33649          SIGND CMP?,SIZE[LONG],          ;
U 1519, 0580,0C10,0B65,8047,0952,9 413* ;33650  NEXT/ED.POD.40                   ;
                                ;33651
                                ;33652          ;11------------------------------; PO = 05-0F, 1X, 2X, 3X, 80
                                ;33653          M[TEMP3]_ZLIT8[0A],             ;
U 151B, 0D86,3DB7,0030,5047,0151,A      ;33654  NEXT/ED.RESV.OPER               ;
                                ;33655
                                ;33656 =01
                                ;33657 ED.POD.40:
                                ;33658          ;01------------------------------; PO = BX
                                ;33659          M[TEMP3]_ZLIT8[0A],             ;
U 1529, 0D86,3DB7,0030,5047,0151,A      ;33660  NEXT/ED.RESV.OPER               ;
                                ;33661
                                ;33662          ;11------------------------------; PO = 81-8F, 9X, AX
                                ;33663          M[TEMP3]_R[TEMP0],              ; SEPERATE FILL, MOVE AND FLOAT
U 152B, 0886,358E,4230,0047,0150,F      ;33664  WB<5-0>?                        ;
                                ;33665
                                ;33666 =001111
                                ;33667          ;001111--------------------------;
                                ;33668          M[TEMP3]_MB.AND.ZLIT0[0F],      ;
U 150F, 0D86,3C12,0030,7847,0155,9      ;33669  NEXT/ED.FILL                    ;
                                ;33670
                                ;33671          ;011111--------------------------;
                                ;33672          M[TEMP3]_MB.AND.ZLIT0[0F],      ;
U 151F, 0586,3C12,0A30,7847,0151,C      ;33673  WX.EQ.0?,NEXT/ED.MOVE           ;
                                ;33674
                                ;33675          ;101111--------------------------;
                                ;33676          M[TEMP3]_MB.AND.ZLIT0[0F],      ;
U 152F, 0586,3C12,0A30,7847,0156,4      ;33677  WX.EQ.0?,NEXT/ED.FLOAT          ;
                                ;33678 =
```

```
                        ;33679 .TOC "  Edit Packed to Character string : PATTERNS 00 - 04"
                        ;33680
                        ;33681 ;*****************************************************************************
                        ;33682 ;       Entry points    ED.00-04
                        ;33683 ;
                        ;33684 ;       Input           MTEMP8          Current PSL
                        ;33685 ;                       R2              Sign'Fill register
                        ;33686 ;                       FLAG0           PSL<C> significance bit
                        ;33687 ;
                        ;33688 ;       Output          TEMP5           Character to be stored
                        ;33689 ;                       MTEMP8          PSL with modified condition codes
                        ;33690 ;                       PSL<C>
                        ;33691 ;                       FLAG0           PSL<C>
                        ;33692 ;
                        ;33693 ;       Subroutines     ED.EXIT
                        ;33694 ;                       ED.LOOP
                        ;33695 ;                       ED.STORE.NOINT
                        ;33696 ;
                        ;33697 ;       This routine processes the following pattern operators:
                        ;33698 ;               00      EO$EXIT
                        ;33699 ;               01      EO$END_FLOAT
                        ;33700 ;               02      EO$CLEAR_SIGNIF
                        ;33701 ;               03      EO$SET_SIGNIF
                        ;33702 ;               04      EO$STORE_SIGN
                        ;33703 ;
                        ;33704 ;       Charlie's flows 6.3
                        ;33705 ;*****************************************************************************
                        ;33706
                        ;33707 .TOC "  Edit Packed to Character string :       EO$END"
                        ;33708 =000
                        ;33709 ED.00-04:
                        ;33710          ;000-----------------------------; EO$END
U 1500, 0C80,0036,4030,0047,0163,3      ;33711          NEXT/ED.EXIT                ;
                        ;33712
                        ;33713 .TOC "  Edit Packed to Character string :       EO$END FLOAT"
                        ;33714          ;001-----------------------------; EO$END FLOAT
                        ;33715          M[TEMP5]_R[R2].RR.8,            ; PREPARE TO STORE SIGN
U 1501, 0886,5287,0404,8047,014F,4      ;33716          FLAG0?,NEXT/ED.END.FLOAT    ; BRANCH ON PSW<C>.
                        ;33717
                        ;33718 .TOC "  Edit Packed to Character string :       EO$CLEAR SIGNIF"
                        ;33719          ;010-----------------------------; EO$CLEAR SIGNIF
                        ;33720          CC_M[TEMP8]_MB.ANDNOT.CONX(1),  ; CLEAR PSW<C>.
                        ;33721          CLEAR FLAG0,                    ;
U 1502, 0006,8713,3000,00A7,014E,2      ;33722          NEXT/ED.LOOP                ;
                        ;33723
                        ;33724 .TOC "  Edit Packed to Character string :       EO$SET SIGNIF"
                        ;33725          ;011-----------------------------; EO$SET SIGNIF
                        ;33726          CC_M[TEMP8]_MB.OR.CONX(1),      ; SET PSW<C>.
                        ;33727          SET FLAG0,                      ;
U 1503, 0C46,8712,4000,00A7,014E,2      ;33728          NEXT/ED.LOOP                ;
```

```
                                ;33729 .TOC "  Edit Packed to Character string :       EO$STORE SIGN"
                                ;33730          ;100-----------------------------; EO$STORE SIGN
                                ;33731          M[TEMP5]_R[R2].RR.8,              ; PREPARE TO STORE SIGN
U 1504, 0486,52B7,0004,8047,0160,8 ;33732       NEXT/ED.STORE.NOINT               ;
                                ;33733 =
                                ;33734 =0
                                ;33735 ED.END.FLOAT:
                                ;33736          ;0-----------------------------;
                                ;33737          CC_M[TEMP8]_MB.OR.CONX(1),        ; PSW<C> IS CLEAR,
                                ;33738          SET FLAG0,                        ; SET IT AND STORE THE SIGN.
U 14F4, 0C46,8712,4000,00A7,0160,8 ;33739       NEXT/ED.STORE.NOINT               ;
                                ;33740
                                ;33741          ;1-----------------------------;
U 14F5, 0480,0036,4030,0047,014E,2 ;33742       NEXT/ED.LOOP                      ; PSW<C> IS SET, DON'T DO ANYTHING
```

```
                        ;33743 .TOC "  Edit Packed to Character string : PATTERNS 40 - 47"
                        ;33744
                        ;33745 ;*****************************************************************************
                        ;33746 ;       Entry points    ED. 40-47
                        ;33747 ;       Input           TEMP0           Pattern operator
                        ;33748 ;                       TEMP6           Byte operand
                        ;33749 ;                       R2              Sign'Fill register
                        ;33750 ;
                        ;33751 ;       Resources       D               Temporary
                        ;33752 ;
                        ;33753 ;       Output          R2              Sign'Fill register
                        ;33754 ;
                        ;33755 ;       Subroutines     ED.LOOP
                        ;33756 ;                       ED.INSERT
                        ;33757 ;                       ED.BLANK.ZERO
                        ;33758 ;                       ED.REPLACE.SIGN
                        ;33759 ;                       ED.ADJUST.INPUT
                        ;33760 ;
                        ;33761 ;       This routine processes the following pattern operators:
                        ;33762 ;                       40      EO$LOAD_FILL
                        ;33763 ;                       41      EO$LOAD_SIGN
                        ;33764 ;                       42      EO$LOAD_;PLUS
                        ;33765 ;                       43      EO$LOAD_MINUS
                        ;33766 ;                       44      EO$INSERT
                        ;33767 ;                       45      EO$BLANK_ZERO
                        ;33768 ;                       46      EO$REPLACE_SIGN
                        ;33769 ;                       47      EO$ADJUST_INPUT
                        ;33770 ;
                        ;33771 ;       Charlie's flows 6.4
                        ;33772 ;*****************************************************************************
                        ;33773
                        ;33774 ED.40-47:
                        ;33775          ;-------------------------------;
                        ;33776          WB_R[TEMP0],                    ;
U 1598, 0080,05BE,4230,0047,0153,8      ;33777          WB<5-0>?                        ; SEPERATE 40,41,42,43,44,45,46, AND 47
                        ;33778
                        ;33779 .TOC "  Edit Packed to Character string :       EO$LOAD FILL"
                        ;33780
                        ;33781 =111000 ;111000-------------------------; EO$LOAD FILL
                        ;33782          R[R2].SIZ_M[TEMP6],             ; LOAD FILL BYTE.
U 1538, 0482,6592,4004,8047,014E,2      ;33783          SIZE[BYTE],NEXT/ED.LOOP         ;
                        ;33784
                        ;33785 .TOC "  Edit Packed to Character string :       EO$LOAD SIGN"
                        ;33786
                        ;33787          ;111001-------------------------; EO$LOAD SIGN
U 1539, 0880,05BE,6034,8047,014F,8      ;33788          D_R[R2],NEXT/ED.LOAD.SIGN       ; MOVE SIGN'FILL REGISTER INTO DREG
```

```
                                ;33789 .TOC "  Edit Packed to Character string :       EO$LOAD PLUS"
                                ;33790
                                ;33791          ;111010-------------------------; EO$LOAD PLUS
                                ;33792          D_R[R2],                         ; MOVE SIGN'FILL REGI TER INTO DREG
U 153A, 0C80,05BE,64F4,8047,014F,8 ;33793       F[AG3?,NEXT/ED.LOAD.SIGN         ;
                                ;33794
                                ;33795 .TOC "  Edit Packed to Character string :       EO$LOAD MINUS"
                                ;33796
                                ;33797          ;111011-------------------------; EO$LOAD MINUS
                                ;33798          D_R[R2],                         ; MOVE SIGN'FILL REGISTER INTO DREG
U 153B, 0480,05BE,64F4,8047,014F,C ;33799       F[AG3?,NEXT/ED.NOT.MINUS         ;
                                ;33800
                                ;33801          ;111100-------------------------; EO$INSERT
U 153C, 0880,003C,4430,0047,0150,C ;33802       FLAG0?,NEXT/ED.INSERT            ; BRANCH ON SIGNIFICANCE
                                ;33803
                                ;33804          ;111101-------------------------; EO$BLANK ZERO
                                ;33805          R[R5]_R8-M[TEMP6],               ; BACK UP DESTINATION ADDRESS
U 153D, 0884,6003,04B5,4047,0151,1 ;33806       FLAG2?,NEXT/ED.BLANK.ZERO        ;
                                ;33807
                                ;33808          ;111110-------------------------; EO$REPLACE SIGN
                                ;33809          WB_M[TEMP8],                     ; BRANCH ON PSL<N,Z>
U 153E, 0880,8592,4230,0047,0157,B ;33810       WB<5-0>?,NEXT/ED.REPLACE.SIGN    ;
                                ;33811
                                ;33812          ;111111-------------------------; EO$ADJUST INPUT
                                ;33813          M[TEMP3]_D_R[R0],                ;
U 153F, 0C86,35BE,6034,0047,015B,B ;33814       NEXT/ED.ADJUST.INPUT             ;
                                ;33815
                                ;33816 =0
                                ;33817 ED.LOAD.SIGN:
                                ;33818          ;0------------------------------;
                                ;33819          D_D.AND.ZLITO[OFF],              ; CLEAR OLD SIGN BYTE
U 14F8, 0D80,0C32,2037,F847,0159,A ;33820       NEXT/ED.SAVE.NEW.SIGN            ;
                                ;33821
                                ;33822          ;1------------------------------;
U 14F9, 0480,0036,4030,0047,014E,2 ;33823       NEXT/ED.LOOP                     ; EO$LOAD PLUS, BUT SOURCE IS MINUS
                                ;33824
                                ;33825 =0
                                ;33826 ED.NOT.MINUS:
                                ;33827          ;0------------------------------;
U 14FC, 0480,0036,4030,0047,014E,2 ;33828       NEXT/ED.LOOP                     ; EO$LOAD MINUS, BUT SOURCE IS PLUS
                                ;33829
                                ;33830          ;1------------------------------;
U 14FD, 0D80,0C32,2037,F847,0159,A ;33831       D_D.AND.ZLITO[OFF]               ; CLEAR OLD SIGN BYTE
                                ;33832
                                ;33833 ED.SAVE.NEW.SIGN:
                                ;33834          ;-------------------------------;
                                ;33835          R[R2]_D.OR.(M[TEMP6],RL,8),      ;
U 159A, 0484,63B2,4024,8047,014E,2 ;33836       NEXT/ED.LOOP                     ;
```

B	1	MICRO2	1M(01)	J	5	MICRO2	1M(01)	E	10	MICRO2	1M(01)
C	1	MICRO2	1M(01)	K	5	MICRO2	1M(01)	F	10	MICRO2	1M(01)
D	1	MICRO2	1M(01)	L	5	MICRO2	1M(01)	G	10	MICRO2	1M(01)
E	1	MICRO2	1M(01)	M	5	MICRO2	1M(01)	H	10	MICRO2	1M(01)
F	1	MICRO2	1M(01)	N	5	MICRO2	1M(01)	I	10	MICRO2	1M(01)
G	1	MICRO2	1M(01)	B	6	MICRO2	1M(01)	J	10	MICRO2	1M(01)
H	1	MICRO2	1M(01)	C	6	MICRO2	1M(01)	K	10	MICRO2	1M(01)
I	1	MICRO2	1M(01)	D	6	MICRO2	1M(01)	L	10	MICRO2	1M(01)
J	1	MICRO2	1M(01)	E	6	MICRO2	1M(01)	M	10	MICRO2	1M(01)
K	1	MICRO2	1M(01)	F	6	MICRO2	1M(01)	N	10	MICRO2	1M(01)
L	1	MICRO2	1M(01)	G	6	MICRO2	1M(01)	B	11	MICRO2	1M(01)
M	1	MICRO2	1M(01)	H	6	MICRO2	1M(01)	C	11	MICRO2	1M(01)
N	1	MICRO2	1M(01)	I	6	MICRO2	1M(01)	D	11	MICRO2	1M(01)
B	2	MICRO2	1M(01)	J	6	MICRO2	1M(01)	E	11	MICRO2	1M(01)
C	2	MICRO2	1M(01)	K	6	MICRO2	1M(01)	F	11	MICRO2	1M(01)
D	2	MICRO2	1M(01)	L	6	MICRO2	1M(01)	G	11	MICRO2	1M(01)
E	2	MICRO2	1M(01)	M	6	MICRO2	1M(01)	H	11	MICRO2	1M(01)
F	2	MICRO2	1M(01)	N	6	MICRO2	1M(01)	I	11	MICRO2	1M(01)
G	2	MICRO2	1M(01)	B	7	MICRO2	1M(01)	J	11	MICRO2	1M(01)
H	2	MICRO2	1M(01)	C	7	MICRO2	1M(01)	K	11	MICRO2	1M(01)
I	2	MICRO2	1M(01)	D	7	MICRO2	1M(01)	L	11	MICRO2	1M(01)
J	2	MICRO2	1M(01)	E	7	MICRO2	1M(01)	M	11	MICRO2	1M(01)
K	2	MICRO2	1M(01)	F	7	MICRO2	1M(01)	N	11	MICRO2	1M(01)
L	2	MICRO2	1M(01)	G	7	MICRO2	1M(01)	B	12	MICRO2	1M(01)
M	2	MICRO2	1M(01)	H	7	MICRO2	1M(01)	C	12	MICRO2	1M(01)
N	2	MICRO2	1M(01)	I	7	MICRO2	1M(01)	D	12	MICRO2	1M(01)
B	3	MICRO2	1M(01)	J	7	MICRO2	1M(01)	E	12	MICRO2	1M(01)
C	3	MICRO2	1M(01)	K	7	MICRO2	1M(01)	F	12	MICRO2	1M(01)
D	3	MICRO2	1M(01)	L	7	MICRO2	1M(01)	G	12	MICRO2	1M(01)
E	3	MICRO2	1M(01)	M	7	MICRO2	1M(01)	H	12	MICRO2	1M(01)
F	3	MICRO2	1M(01)	N	7	MICRO2	1M(01)	I	12	MICRO2	1M(01)
G	3	MICRO2	1M(01)	B	8	MICRO2	1M(01)	J	12	MICRO2	1M(01)
H	3	MICRO2	1M(01)	C	8	MICRO2	1M(01)	K	12	MICRO2	1M(01)
I	3	MICRO2	1M(01)	D	8	MICRO2	1M(01)	L	12	MICRO2	1M(01)
J	3	MICRO2	1M(01)	E	8	MICRO2	1M(01)	M	12	MICRO2	1M(01)
K	3	MICRO2	1M(01)	F	8	MICRO2	1M(01)	N	12	MICRO2	1M(01)
L	3	MICRO2	1M(01)	G	8	MICRO2	1M(01)	B	13	MICRO2	1M(01)
M	3	MICRO2	1M(01)	H	8	MICRO2	1M(01)	C	13	MICRO2	1M(01)
N	3	MICRO2	1M(01)	I	8	MICRO2	1M(01)	D	13	MICRO2	1M(01)
B	4	MICRO2	1M(01)	J	8	MICRO2	1M(01)	E	13	MICRO2	1M(01)
C	4	MICRO2	1M(01)	K	8	MICRO2	1M(01)	F	13	MICRO2	1M(01)
D	4	MICRO2	1M(01)	L	8	MICRO2	1M(01)	G	13	MICRO2	1M(01)
E	4	MICRO2	1M(01)	M	8	MICRO2	1M(01)	H	13	MICRO2	1M(01)
F	4	MICRO2	1M(01)	N	8	MICRO2	1M(01)	I	13	MICRO2	1M(01)
G	4	MICRO2	1M(01)	B	9	MICRO2	1M(01)	J	13	MICRO2	1M(01)
H	4	MICRO2	1M(01)	C	9	MICRO2	1M(01)	K	13	MICRO2	1M(01)
I	4	MICRO2	1M(01)	D	9	MICRO2	1M(01)	L	13	MICRO2	1M(01)
J	4	MICRO2	1M(01)	E	9	MICRO2	1M(01)	M	13	MICRO2	1M(01)
K	4	MICRO2	1M(01)	F	9	MICRO2	1M(01)	N	13	MICRO2	1M(01)
L	4	MICRO2	1M(01)	G	9	MICRO2	1M(01)	B	14	MICRO2	1M(01)
M	4	MICRO2	1M(01)	H	9	MICRO2	1M(01)	C	14	MICRO2	1M(01)
N	4	MICRO2	1M(01)	I	9	MICRO2	1M(01)	D	14	MICRO2	1M(01)
B	5	MICRO2	1M(01)	J	9	MICRO2	1M(01)	E	14	MICRO2	1M(01)
C	5	MICRO2	1M(01)	K	9	MICRO2	1M(01)	F	14	MICRO2	1M(01)
D	5	MICRO2	1M(01)	L	9	MICRO2	1M(01)	G	14	MICRO2	1M(01)
E	5	MICRO2	1M(01)	M	9	MICRO2	1M(01)	H	14	MICRO2	1M(01)
F	5	MICRO2	1M(01)	N	9	MICRO2	1M(01)	I	14	MICRO2	1M(01)
G	5	MICRO2	1M(01)	B	10	MICRO2	1M(01)	J	14	MICRO2	1M(01)
H	5	MICRO2	1M(01)	C	10	MICRO2	1M(01)	K	14	MICRO2	1M(01)
I	5	MICRO2	1M(01)	D	10	MICRO2	1M(01)	L	14	MICRO2	1M(01)

M	14	MICRO2	1M(01)
N	14	MICRO2	1M(01)
B	15	MICRO2	1M(01)
C	15	MICRO2	1M(01)
D	15	MICRO2	1M(01)
E	15	MICRO2	1M(01)
F	15	MICRO2	1M(01)
G	15	MICRO2	1M(01)
H	15	MICRO2	1M(01)
I	15	MICRO2	1M(01)
J	15	MICRO2	1M(01)
K	15	MICRO2	1M(01)
L	15	MICRO2	1M(01)
M	15	MICRO2	1M(01)
N	15	MICRO2	1M(01)
B	16	MICRO2	1M(01)
C	16	MICRO2	1M(01)
D	16	MICRO2	1M(01)
E	16	MICRO2	1M(01)
F	16	MICRO2	1M(01)
G	16	MICRO2	1M(01)
H	16	MICRO2	1M(01)
I	16	MICRO2	1M(01)
J	16	MICRO2	1M(01)
K	16	MICRO2	1M(01)
L	16	MICRO2	1M(01)

```
                        ;33837 .TOC "  Edit Packed to Character string :       EO$INSERT"
                        ;33838
                        ;33839 ;*****************************************************************************
                        ;33840 ;       Entry points    ED.INSERT
                        ;33841 ;       Input           TEMP6           Character operand
                        ;33842 ;                       R2              Sign'Fill register
                        ;33843 ;
                        ;33844 ;       Output          TEMP5           Character to be inserted
                        ;33845 ;
                        ;33846 ;       Subroutines     ED.STORE.NOINT
                        ;33847 ;
                        ;33848 ;       This routine processes the following pattern operator:
                        ;33849 ;
                        ;33850 ;                       44      EO$INSERT
                        ;33851 ;
                        ;33852 ;       Charlie's flows 6.5
                        ;33853 ;*****************************************************************************
                        ;33854
                        ;33855 =0
                        ;33856 ED.INSERT:
                        ;33857         ;0-------------------------------;
                        ;33858         M[TEMP5],R[R2],                 ; SIGNIFICANCE IS NOT SET, STORE FILL
U 150C, 0486,55BE,4034,8047,0160,8      ;33859         NEXT/ED.STORE.NOINT             ;
                        ;33860
                        ;33861         ;1-------------------------------;
                        ;33862         M[TEMP5],R[TEMP6],              ; SIGNIFICANCE IS SET, STORE CHARACTER
U 150D, 0486,55BE,4031,8047,0160,8      ;33863         NEXT/ED.STORE.NOINT             ;
```

```
                        ;33864 .TOC "  Edit Packed to Character string :       EO$BLANK ZERO"
                        ;33865
                        ;33866 ;*****************************************************************************
                        ;33867 ;       Entry points    ED.BLANK.ZERO
                        ;33868 ;
                        ;33869 ;       Input           TEMP6           Length operand
                        ;33870 ;                       R2              Sign'Fill register
                        ;33871 ;                       R5              Destination address
                        ;33872 ;
                        ;33873 ;       Resources       TEMP6           Counter for blanking with fill
                        ;33874 ;                       TEMP5           Pass fill character to ED.STORE
                        ;33875 ;
                        ;33876 ;       Output          TEMP3           FPD return code
                        ;33877 ;
                        ;33878 ;       Subroutines     ED.LOOP
                        ;33879 ;                       ED.STORE
                        ;33880 ;                       ED.PACK
                        ;33881 ;
                        ;33882 ;       This routine processes the following pattern operators:
                        ;33883 ;
                        ;33884 ;                       45      EO$BLANK_ZERO
                        ;33885 ;
                        ;33886 ;       Charlie's flows 6.5
                        ;33887 ;*****************************************************************************
                        ;33888
                        ;33889 =000
                        ;33890 =001
                        ;33891 ED.BLANK.ZERO:
                        ;33892         ;001-----------------------------;
                        ;33893         R[R5]_M[TEMP6]+RB,              ; UNDO PREVIOUS MICRO INSTRUCTION
U 1511, 0484,6001,0035,4047,014E,2      ;33894         NEXT/ED.LOOP                    ; IF VALUE IS NOT ZERO.
                        ;33895 =100
                        ;33896 ED.BLANK.ZERO.LOOP:
                        ;33897         ;100-----------------------------;
                        ;33898         M[TEMP5]_R[R2],                 ; STORE FILL
U 1514, 0486,55BE,4034,8047,0560,6      ;33899         PUSH,NEXT/ED.STORE              ;
                        ;33900
                        ;33901         ;101-----------------------------; RETURN + 1 FROM ED.STORE
                        ;33902         M[TEMP6]_MB-ZLIT0[1],           ; DECREMENT COUNT OF BYTES TO BLANK OUT.
                        ;33903         WB<31-30S?,                     ; CHECK IF NEGATIVE
U 1515, 0986,6C10,06F0,0847,0951,4 377* ;33904         NEXT/ED.BLANK.ZERO.LOOP         ;
                        ;33905
                        ;33906         ;110-----------------------------; RETURN + 2 FROM ED.STORE
                        ;33907         M[TEMP3]_R[TEMP6],              ; FPD PACK RETURN FROM ED.STORE,
U 1516, 0486,35BE,4031,8047,0159,E      ;33908         NEXT/ED.BLANK.ZERO.PACK         ; SAVE COUNT AND PACK IT UP.
                        ;33909
                        ;33910
                        ;33911         ;111-----------------------------;
U 1517, 0480,0036,4030,0047,014E,2      ;33912         NEXT/ED.LOOP                    ; DONE
                        ;33913
                        ;33914 ED.BLANK.ZERO.PACK:
                        ;33915         ;-------------------------------;
                        ;33916         M[TEMP3]_MB.OR.ZLIT8[8],        ; SAVE PACK CODE.
U 159E, 0186,3D92,4030,4047,0165,0      ;33917         NEXT/ED.PACK                    ;
```

```
                        ;33918 .TOC "  Edit Packed to Character string :       EO$REPLACE SIGN"
                        ;33919
                        ;33920 ;*****************************************************************************
                        ;33921 ;       Entry points    ED.REPLACE.SIGN
                        ;33922 ;
                        ;33923 ;       Input           TEMP6           Length operand
                        ;33924 ;                       R5              Destination address
                        ;33925 ;
                        ;33926 ;       Output          VA              Address of destination sign
                        ;33927 ;                       TEMP5           Pass fill to ED.STORE.NOINT.10
                        ;33928 ;
                        ;33929 ;       Subroutines     ED.STORE.NOINT.10
                        ;33930 ;                       ED.LOOP
                        ;33931 ;
                        ;33932 ;       This routine processes the following pattern operator:
                        ;33933 ;
                        ;33934 ;                       46      EO$REPLACE_SIGN
                        ;33935 ;
                        ;33936 ;       Charlie's flows 6.6
                        ;33937 ;*****************************************************************************
                        ;33938
                        ;33939 =111011
                        ;33940 ED.REPLACE.SIGN:
                        ;33941         ;111011-------------------------;
U 157B, 0480,0036,4030,0047,014E,2      ;33942         NEXT/ED.LOOP                    ; DESTINATION IS NON-ZERO, DO NOTHING.
                        ;33943
                        ;33944         ;111111-------------------------; DESTINATION IS 0, CHANGE SIGN TO FILL
U 157F, 0480,6003,0035,44A7,015A,B      ;33945         VA_R[R5]-M[TEMP6]               ; LOAD ADDRESS OF DESTINATION SIGN
                        ;33946
                        ;33947         ;-------------------------------; ED.STORE.NOINT.10 WILL ADD 1 TO R5
U 15AB, 0484,0E7D,0035,4047,00BF,2      ;33948         R[R5]_RB-1                      ; FOR A NET EFFECT OF NO CHANGE.
                        ;33949
                        ;33950 0BF2:    ;*********FORCE ADDRESS*********;
                        ;33951         ;0**-----------------------------;
                        ;33952         WRITE R[R2],SIZE[BYTE],         ; STORE FILL
U 0BF2, 0C80,05BE,4004,85D8,0161,4      ;33953         NEXT/ED.STORE.NOINT.DONE        ;
                        ;33954
                        ;33955 0BF6:    ;1** ****FORCE ADDRESS**********; FPD RETURN FROM UTRAPPED WRITE
U 0BF6, 0581,AC10,0030,0C87,015A,E      ;33956         PC_M[PC]-ZLIT0[1]               ; FIXUP PC FOR PACKING
                        ;33957
                        ;33958         ;-------------------------------;
                        ;33959         R[R5]_RB+1,                     ; RESTORE R5
U 15AE, 0884,0E7C,0035,4047,014F,7      ;33960         NEXT/ED.POD.16                  ;
```

```
                        ;33961 .TOC "  Edit Packed to Character string :       EO$ADJUST INPUT"
                        ;33962
                        ;33963 ;*****************************************************************************
                        ;33964 ;       Entry points    ED.ADJUST.INPUT
                        ;33965 ;
                        ;33966 ;       Input           TEMP6           Length operand
                        ;33967 ;                       D               Source length
                        ;33968 ;
                        ;33969 ;       Resources       TEMP3           Count of characters to discard
                        ;33970 ;                       TEMP5           Input character from ED.READ
                        ;33971 ;
                        ;33972 ;       Output          R0              Adjusted source length
                        ;33973 ;                       PSL<V,C>
                        ;33974 ;                       TEMP8           Modified PSL
                        ;33975 ;                       FLAG0           PSL<C>
                        ;33976 ;                       FLAG1           PSL<V>
                        ;33977 ;                       TEMP3           FPD return code'count of chars to disgar
                        ;33978 ;
                        ;33979 ;       Subroutines     ED.LOOP
                        ;33980 ;                       ED.READ
                        ;33981 ;                       ED.MOVE.PACK
                        ;33982 ;
                        ;33983 ;       This routine processes the following pattern operator:
                        ;33984 ;
                        ;33985 ;                       47      EO$ADJUST_INPUT
                        ;33986 ;
                        ;33987 ;       Charlie's flows 6.7
                        ;33988 ;*****************************************************************************
                        ;33989
                        ;33990 ED.ADJUST.INPUT:
                        ;33991                 ;-------------------------------;
                        ;33992                 M[TEMP3]_ZEXT(MB)-R[TEMP6],     ;
                        ;33993                 SIZE[WORD],WB<31-30>?,          ;
U 158B, 0C86,3014,06D1,3047,0954,9 349*         ;33994                 NEXT/ED.ADJUST.LOOP             ;
                        ;33995 =00
                        ;33996 ED.ADJUST.OVERFLOW:
                        ;33997                 ;00-----------------------------;
                        ;33998                 M[TEMP8]_MB.OR.ZLIT0[3],        ; SET PSL<V,C>
U 1548, 0186,8C12,4030,1847,015C,A              ;33999                 NEXT/ED.ADJUST.SET.CC           ;
                        ;34000
                        ;34001 ED.ADJUST.LOOP:
                        ;34002                 ;01-----------------------------; SOURCE LENGTH IS GREATER THAN
                        ;34003                 M[TEMP3]_MB-ZLIT0[1],           ; DESTINATION LENGTH,
                        ;34004                 WB<31-30>?,                     ; DISCARD EXTRA INPUT.
U 1549, 0986,3C10,06F0,0847,0952,1 377*         ;34005                 NEXT/ED.ADJUST.P1               ;
                        ;34006
                        ;34007 =11              ;11-----------------------------; DESTINATION LENGTH IS GREATER
U 154B, 0C84,33B7,0014,0047,015B,E              ;34008                 R[R0]_M[TEMP3].RR.16            ; THAN INPUT LENGTH
                        ;34009
                        ;34010                 ;-------------------------------; LOAD COUNT OF EXTRA ZEROS
                        ;34011                 R[R0].SIZ_D,                    ; TO RETURN FROM READ ROUTINE.
U 15BE, 0082,05B2,4014,0047,014E,2              ;34012                 SIZE[WORD],NEXT/ED.LOOP         ;
```

```
                                    ;34013 =000
                                    ;34014 =001
                                    ;34015 ED.ADJUST.P1:
                                    ;34016          ;001-------------------------------;
U 1521, 0C80,0036,4030,0U47,055F,2  ;34017          PUSH,NEXT/ED.READ                 ;
                                    ;34018
                                    ;34019          ;010-------------------------------;
                                    ;34020          WB_M[TEMP5],                      ;
                                    ;34021          WX.EQ.0?,                         ;
U 1522, 0080,5592,4A30,0047,0954,8 322* ;34022      NEXT/ED.ADJUST.OVERFLOW           ;
                                    ;34023
                                    ;34024          ;011-------------------------------;
U 1523, 0480,0036,4030,0047,014E,2  ;34025          NEXT/ED.LOOP                      ;
                                    ;34026
                                    ;34027 =101     ;101-------------------------------;
                                    ;34028          M[TEMP3]_MB.OR.ZLIT8[2],          ;
U 1525, 0986,3D92,4030,1047,015D,0  ;34029          NEXT/ED.MOVE.PACK                 ;
                                    ;34030 =
                                    ;34031 ED.ADJUST.SET.CC:
                                    ;34032          ;-------------------------------;
U 15CA, 0980,8C1?,0030,7B07,015C,E  ;34033          FLAGS_M[TEMP8].AND.ZLIT0[0F]      ; SET PSL<V,C>
                                    ;34034
                                    ;34035          ;-------------------------------;
                                    ;34036          CC_M[TEMP8]_MB.ANDNOT.CONX(4),    ;
                                    ;34037          CLEAR FLAG2,                      ; CLEAR PSL<Z>
U 15CE, 0C16,8713,8020,00A7,0154,9  ;34038          NEXT/ED.ADJUST.LOOP               ;
```

```
                                ;34039 .TOC "  Edit Packed to Character string : PATTERNS  81 - AF"
                                ;34040 .TOC "  Edit Packed to Character string :       EO$FILL"
                                ;34041
                                ;34042 ;*****************************************************************************
                                ;34043 ;       Entry points    ED.FILL
                                ;34044 ;
                                ;34045 ;       Input           TEMP3           Count operand
                                ;34046 ;                       R2              Sign'Fill register
                                ;34047 ;
                                ;34048 ;       Resources       TEMP5           Pass fill character to ED.STORE
                                ;34049 ;                       TEMP3           Loop counter
                                ;34050 ;
                                ;34051 ;       Output          TEMP3           FPD return code
                                ;34052 ;
                                ;34053 ;       Subroutines     ED.STORE
                                ;34054 ;                       ED.PACK
                                ;34055 ;                       ED.LOOP
                                ;34056 ;
                                ;34057 ;       This routine processes the following pattern operator:
                                ;34058 ;
                                ;34059 ;                       8X      EO$FILL
                                ;34060 ;
                                ;34061 ;       Charlie's flows 6.8
                                ;34062 ;*****************************************************************************
                                ;34063 =00
                                ;34064 =01
                                ;34065 ED.FILL:
                                ;34066          ;01-------------------------------;
                                ;34067          M[TEMP5]_R[R2],                 ; STORE FILL
U 1559, 0486,55BE,4034,8047,0560,6      ;34068          PUSH,NEXT/ED.STORE              ;
                                ;34069
                                ;34070          ;10-------------------------------;
                                ;34071          M[TEMP3]_MB-ZLIT0[1],           ; DECREMENT COUNT OF FILL CHARACTERS
U 155A, 0586,3C10,0A70,0847,0956,8 378* ;34072          WX.NE.0?,NEXT/ED.FILL.LOOP      ;
                                ;34073
                                ;34074          ;11-------------------------------;
                                ;34075          M[TEMP3]_MB.OR.ZLIT8[6],        ;
U 155B, 0986,3D92,4030,3047,0165,0      ;34076          NEXT/ED.PACK                    ;
                                ;34077 =00
                                ;34078 ED.FILL.LOOP:
                                ;34079          ;00-------------------------------;
U 1568, 0480,0036,4030,0047,014E,2      ;34080          NEXT/ED.LOOP                    ;
                                ;34081
                                ;34082 ED.FILL.P2:
                                ;34083          ;01-------------------------------;
                                ;34084          M[TEMP5]_R[R2],                 ; STORE FILL
U 1569, 0486,55BE,4034,8047,0560,6      ;34085          PUSH,NEXT/ED.STORE              ;
                                ;34086
                                ;34087          ;10-------------------------------; RETURN + 1 FROM ED.SOTRE
                                ;34088          M[TEMP3]_MB-ZLIT0[1],           ; DECREMENT COUNT OF FILL CHARACTERS
U 156A, 0580,3C10,0A70,0847,0956,8 378* ;34089          WX.NE.0?,NEXT/ED.FILL.LOOP      ; CHECK IF DONE
                                ;34090
                                ;34091          ;11-------------------------------; RETURN + 2 FROM ED.STORE
                                ;34092          M[TEMP3]_MB.OR.ZLIT8[7],        ; FPD PACK RETURN FROM ED.STORE,
U 156B, 0D86,3D92,4030,3847,0165,0      ;34093          NEXT/ED.PACK                    ; SAVE CODE AND PACK IT UP.
```

```
                                ;34094 .TOC "  Edit Packed to Character string :       EO$MOVE"
                                ;34095
                                ;34096 ;*****************************************************************************
                                ;34097 ;       Entry points    ED.MOVE
                                ;34098 ;
                                ;34099 ;       Input           TEMP3           Count operand
                                ;34100 ;                       MTEMP8          Current PSL
                                ;34101 ;                       R2              Sign'Fill register
                                ;34102 ;                       FLAG0           PSL<C> significance bit
                                ;34103 ;
                                ;34104 ;       Resources       TEMP3           Loop counter
                                ;34105 ;                       TEMP5           Pass character to ED.STORE
                                ;34106 ;
                                ;34107 ;       Ourout          TEMP1           (Last digit read)'TEMP1<7:0>
                                ;34108 ;                       TEMP3           (FPD return code)'(loop counter)
                                ;34109 ;                       R0              Corrected source length if interrupted
                                ;34110 ;                       MTEMP8          Current PSL
                                ;34111 ;                       PSL<C>          Set if non zero digit moved
                                ;34112 ;                       FLAG0           PSL<C>
                                ;34113 ;
                                ;34114 ;       Subroutines     ED.STORE
                                ;34115 ;                       ED.READ
                                ;34116 ;                       ED.RESV.OPER
                                ;34117 ;                       ED.PACK
                                ;34118 ;                       ED.LOOP
                                ;34119 ;
                                ;34120 ;       This routine processes the following pattern operator:
                                ;34121 ;
                                ;34122 ;                       9X      EO$MOVE
                                ;34123 ;
                                ;34124 ;       Charlie's flows 6.9
                                ;34125 ;***************************************************************************** *********
                                ;34126 =0
                                ;34127 ED.MOVE:
                                ;34128         ;0------------------------------;
                                ;34129         NEXT/ED.MOVE.P1                 ;
                                ;34130
                                ;34131         ;1------------------------------;
                                ;34132         M[TEMP3]_MB.OR.ZLIT8[0A],       ;
U 151D, 0D86,3D92,4030,5047,0151,A  ;34133     NEXT/ED.RESV.OPER               ;
                                ;34134 =000
                                ;34135 ED.MOVE.LOOP:
                                ;34136         ;000----------------------------;
U 1530, 0480,0036,4030,0047,014E,2  ;34137     NEXT/ED.LOOP                    ;
                                ;34138
                                ;34139 ED.MOVE.P1:
                                ;34140         ;001----------------------------;
U 1531, 0C80,0036,4030,0047,055F,2  ;34141     PUSH,NEXT/ED.READ               ;
                                ;34142
                                ;34143 ED.MOVE.P2:
                                ;34144         ;010----------------------------;
                                ;34145         WB_M[TEMP5].AND.ZLIT0[0F],      ;
                                ;34146         PUSH,                           ; PUSH FOR JUMP TO ED.STORE
U 1532, 0580,5C12,0A30,7847,0555,0  ;34147     WX.EQ.07,NEXT/ED.MOVE.SIGNIF    ;
```

```
                                    ;34148          ;011-------------------------------; RETURN + 1 FROM ED.STORE
                                    ;34149          M[TEMP3]_MB-ZLIT0[1],             ; DECREMENT COUNT OF DIGITS TO MOVE
U 1533, 0D86,3C10,0A70,0847,0953,0 378* ;34150      WX.NE.0?,NEXT/ED.MOVE.LOOP        ; CHECK IF DONE
                                    ;34151
                                    ;34152          ;100-------------------------------; RETURN + 2 FROM ED.STORE
                                    ;34153          M[TEMP3]_MB.OR.ZLIT8[3],          ; FPD PACK RETURN, SAVE CODE
U 1534, 0986,3D92,4430,1847,0152,C  ;34154          FLAG0?,NEXT/ED.MOVE.UNCONVERT     ;  CHECK IF STORING DIGIT OR FILL
                                    ;34155
                                    ;34156          ;101-------------------------------; RETURN + 4 FROM ED.READ
U 1535, 0D8C,3D92,4030,0047,015D,0  ;34157          M[TEMP3]_MB.OR.ZLIT8[0]           ; SAVE CODE AND PACK IT UP.
                                    ;34158 =
                                    ;34159 ED.MOVE.PACK:
                                    ;34160          ;--------------------------------;
                                    ;34161          R[R0]_R8+1,                       ;
U 15D0, 0R84,0E7C,0034,0047,0165,0  ;34162          NEXT/ED.PACK                      ;
                                    ;34163
                                    ;34164 =0
                                    ;34165 ED.MOVE.UNCONVERT:
                                    ;34166          ;0-------------------------------;
                                    ;34167          M[TEMP1]_MB.AND.ZLIT0[OFF],       ; SAVE A ZERO
U 152C, 0986,1C12,0037,F847,01C5,0  ;34168          NEXT/ED.PACK                      ;
                                    ;34169
                                    ;34170          ;1-------------------------------;
                                    ;34171          M[TEMP1]_MB-ZLIT8[30],            ; SAVE UNCONVERTED CHARACTER
U 152D, 0186,1D90,0031,8047,0165,0  ;34172          NEXT/ED.PACK                      ;
                                    ;34173
                                    ;34174 =0
                                    ;34175 ED.MOVE.SIGNIF:
                                    ;34176          ;0-------------------------------;
                                    ;34177          M[TEMP8]_MB.OR.ZLIT0[1],          ;
U 1550, 0546,8C12,4030,0847,015D,2  ;34178          SET FLAG0,NEXT/ED.MOVE.CLEAR.Z    ;
                                    ;34179
                                    ;34180          ;1-------------------------------;
U 1551, 0880,0036,4430,0047,0156,0  ;34181          FLAG0?,NEXT/ED.MOVE.STORE         ;
                                    ;34182
                                    ;34183 ED.MOVE.CLEAR.Z:
                                    ;34184          ;--------------------------------;
                                    ;34185          CC_M[TEMP8]_MB.ANDNOT.CONX(4),    ;
                                    ;34186          CLEAR FLAG2,                      ; CLEAR PSL<Z>
U 15D2, 0C16,8713,8020,00A7,0156,1  ;34187          NEXT/ED.MOVE.STORE.DIGIT          ;
                                    ;34188
                                    ;34189 =0
                                    ;34190 ED.MOVE.STORE:
                                    ;34191          ;0-------------------------------;
                                    ;34192          M[TEMP5]_R[R2],                   ; STORE FILL
U 1560, 0C86,558E,403[illegible],8047,0160,6  ;34193  NEXT/ED.STORE                    ;
                                    ;34194
                                    ;34195 ED.MOVE.STORE.DIGIT:
                                    ;34196          ;1-------------------------------;
                                    ;34197          M[TEMP5]_MB+ZLIT0[30],            ; STORE CONVERTED SOURCE DIGIT
U 1561, 0186,5C11,0031,8047,0160,6  ;34198          NEXT/ED.STORE                     ;
```

```
                                                                J  1
                        ;34199 .TOC "  Edit Packed to Character string :       EO$FLOAT"
                        ;34200
                        ;34201 ;*****************************************************************************
                        ;34202 ;       Entry points    ED.FLOAT
                        ;34203 ;
                        ;34204 ;       Input           TEMP3           Count operand
                        ;34205 ;                       MTEMP8          Current PSL
                        ;34206 ;                       R2              Sign'Fill register
                        ;34207 ;                       FLAG0           PSL<C> significance bit
                        ;34208 ;
                        ;34209 ;       Resources       TEMP3           Loop counter
                        ;34210 ;                       TEMP5           Pass character to ED.STORE
                        ;34211 ;                       MTEMP10         Save digit read while storing sign
                        ;34212 ;
                        ;34213 ;       Output          TEMP1           (Last digit read)'TEMP1<7:0>
                        ;34214 ;                       TEMP3           (FPD return code)'(loop counter)
                        ;34215 ;                       MTEMP8          Current PSL
                        ;34216 ;                       PSL<C>          Set if non zero digit moved
                        ;34217 ;                       FLAG0           PSL<C>
                        ;34218 ;
                        ;34219 ;       Subroutines     ED.STORE
                        ;34220 ;                       ED.READ
                        ;34221 ;                       ED.RESV.OPER
                        ;34222 ;                       ED.PACK
                        ;34223 ;                       ED.LOOP
                        ;34224 ;                       ED.MOVE.UNCONVERT
                        ;34225 ;                       ED.MOVE.PACK
                        ;34226 ;
                        ;34227 ;       This routine processes the following pattern operator:
                        ;34228 ;
                        ;34229 ;                               9X      EO$FLOAT
                        ;34230 ;
                        ;34231 ;       Charlie's flows 6.10
                        ;34232 ;*****************************************************************************
                        ;34233
                        ;34234 =0
                        ;34235 ED.FLOAT:
                        ;34236          ;0------------------------------;
U 1564, 0C80,0036,4030,0047,0154,1      ;34237          NEXT/ED.FLOAT.P1                ;
                        ;34238
                        ;34239          ;1------------------------------;
                        ;34240          M[TEMP3] MB.OR.ZLIT8[0A],       ;
U 1565, 0D86,3D92,4030,5047,0151,A      ;34241          NEXT/ED.RESV.OPER               ;
```

```
                                ;34242 =000
                                ;34243 ED.FLOAT.LOOP:
                                ;34244          ;000-------------------------------;
U 1540, 0480,0036,4030,0047,014E,2 ;34245       NEXT/ED.LOOP                      ;
                                ;34246
                                ;34247 ED.FLOAT.P1:
                                ;34248          ;001-------------------------------;
U 1541, 0C80,0036,4030,0047,055F,2 ;34249       PUSH,NEXT/ED.READ                 ;
                                ;34250
                                ;34251 ED.FLOAT.P2:
                                ;34252          ;010-------------------------------;
                                ;34253          M[TEMP10]_R[TEMP5]-0,             ;
                                ;34254          PUSH,                             ; PUSH FOR JUMP TO ED.STORE
                                ;34255          SIGND CMP DEF?,SIZE[BYTE],        ; CHECK IF DIGIT IS ZERO
U 1542, 0C86,A58C,CB41,4047,0D51,2 415* ;34256  NEXT/ED.FLOAT.SIGNIF              ;
                                ;34257
                                ;34258          ;011-------------------------------; RETURN + 1 FROM ED.STORE
                                ;34259          M[TEMP3]_MB-ZLIT0[1],             ; DECREMENT COUNT OF DIGITS TO MOVE
U 1543, 0586,3C10,0A70,0847,0954,0 378* ;34260  WX.NE.0?,NEXT/ED.FLOAT.LOOP       ; CHECK IF DONE
                                ;34261
                                ;34262          ;100-------------------------------; RETURN + 2 FROM ED.STORE
                                ;34263          M[TEMP3]_MB.OR.ZLIT8[5],          ; FPD PACK RETURN, SAVE CODE.
U 1544, 0586,3D92,4430,2847,0152,C ;34264       FLAG0?,NEXT/ED.MOVE.UNCONVERT     ; STORING FILL OR DIGIT?
                                ;34265
                                ;34266          ;101-------------------------------; RETURN + 4 FORM ED.READ
                                ;34267          M[TEMP3]_MB.OR.ZLIT8[1],          ; FPD PACK RETURN FROM ED.READ,
U 1545, 0986,3D92,4030,0847,015D,0 ;34268       NEXT/ED.MOVE.PACK                 ; SAVE CODE AND PACK IT UP.
                                ;34269 =
```

```
                                ;34270 =10
                                ;34271 ED.FLOAT.SIGNIF:
                                ;34272          ;10-------------------------------;
                                ;34273          M[TEMP8]_MB.OR.ZLIT0[1],        ;
                                ;34274          SE: FLAG0,                      ;
U 1512, 0946,8C12,4430,0847,0155,2  ;34275      FLAG0?,NEXT/ED.FLOAT.CHK.PSLC   ;
                                ;34276
                                ;34277 ED.FLOAT.10^:
                                ;34278          ;11-------------------------------; RESTORE TEMP5, IT GETS CLOBBERED
                                ;34279          R[TEMP5]_M[TEMP10],             ; IF THE SIGN IS STORED.
U 1513, 0884,A592,4431,4047,0157,4  ;34280      FLAG0?,NEXT/ED.FLOAT.STORE      ;
                                ;34281 =000
                                ;34282 =010
                                ;34283 ED.FLOAT.CHK.PSLC:
                                ;34284          ;010------------------------------; PSL<C> = 0,
                                ;34285          CC_M[TEMP8]_MB.ANDNOT.CONX(4),  ; SIGNIFICANCE WAS CLEAR,
                                ;34286          CLEAR FLAG2,                    ; CLEAR PSL<Z>
U 1552, 0C16,8713,8020,00A7,055E,E  ;34287      PUSH,NEXT/ED.FLOAT.STORE.SIGN   ;
                                ;34288
                                ;34289          ;011------------------------------; RESTORE TEMP5, IT GETS CLOBBERED
                                ;34290          R[TEMP5]_M[TEMP10],             ; IF THE SIGN IS STORED.
U 1553, 0884,A592,4431,4047,0157,4  ;34291      FLAG0?,NEXT/ED.FLOAT.STORE      ;
                                ;34292
                                ;34293          ;100------------------------------; RETURN + 2 FROM ED.STORE
U 1554, 0186,3D92,4030,2847,015D,6  ;34294      M[TEMP3]_MB.OR.ZLIT8[5]         ; FPD PACK RETURN, SAVE CODE.
                                ;34295 =
                                ;34296          ;-------------------------------;
                                ;34297          CC_M[TEMP8]_MB.ANDNOT.CONX(1),  ; CLEAR SIGNIFICANCE BECAUSE
U 15D4, 0806,8713,8000,00A7,015D,E  ;34298      CLEAR FLAG0                     ; IT WILL BE REDONE AFTER RESTART
                                ;34299
                                ;34300          ;-------------------------------;
U 15DE, 0186,1C12,0037,F847,015E,A  ;34301      M[TEMP1]_MB.AND.ZLIT0[0FF]      ; ERASE DIGIT SAVED BY ED.STORE
                                ;34302
                                ;34303          ;-------------------------------;
                                ;34304          R[TEMP1]_RB.OR.(M[TEMP10].RL.8),        ; SAVE DIGIT WHICH IS IN TEMP10
U 15EA, 0884,A3BE,4020,4047,0165,0  ;34305      NEXT/ED.PACK                    ;
                                ;34306
                                ;34307 ED.FLOAT.STORE.SIGN:
                                ;34308          ;-------------------------------;
                                ;34309          M[TEMP5]_R[R2].RR.8,            ; STORE SIGN
U 15EE, 0C86,52B7,0004,8047,0160,6  ;34310      NEXT/ED.STORE                   ;
                                ;34311 =0
                                ;34312 ED.FLOAT.STORE:
                                ;34313          ;0--------------------------------;
                                ;34314          M[TEMP5]_R[R2],                 ; STORE FILL
U 1574, 0C86,55BE,4034,8047,0160,6  ;34315      NEXT/ED.STORE                   ;
                                ;34316
                                ;34317          ;1--------------------------------;
                                ;34318          M[TEMP5]_MB+ZLIT0[30],          ; STORE CONVERTED SOURCE DIGIT
U 1575, 0186,5C11,0031,8047,0160,6  ;34319      NEXT/ED.STORE                   ;
```

```
                                        .TOC "  Edit Packed to Character string : SOURCE READ ROUTINE"

                                        ;*********************************************************************************
                                        ;       Entry points    ED.READ
                                        ;
                                        ;       Input           R0              See below
                                        ;                       R1              Source address
                                        ;
                                        ;       Resources       MTEMP10         Temporary
                                        ;                       PL
                                        ;                       MDR
                                        ;
                                        ;       Output          TEMP5           Zero extended source nibble
                                        ;                       R0              See below
                                        ;                       R1              Source address
                                        ;
                                        ;       Subroutines     ED.RESV.OPER.ABORT
                                        ;                       IE.SERV.IP.TS
                                        ;
                                        ;       This routine reads the next nibble from the source or
                                        ;       returns a zero if R0 is negative.
                                        ;       R0<31:16> contains a negative count of zeros to supply and
                                        ;       R0<15:0> contains the source length remaining.
                                        ;
                                        ;       Charlie's flows 6.11
                                        ;*********************************************************************************

                                        ED.READ:
                                                ;--------------------------------;
                                                M[TEMP10]_R[R(]-0,               ;
                                                SIZE[LONG],                      ;
U 15F2, 0486,A5BC,0B64,0047,0958,8 415*         SIGND CMP?                       ;

                                        =00     ;00------------------------------;
                                                R[R0]_RB-1,                      ; NORMAL READ
U 1588, 0C84,0E7D,0274,0047,0952,6 371*         WB<1-0>?,NEXT/ED.READ.10         ;

                                                ;01------------------------------;
                                                R[R0]_RB-1,                      ; LENGTH = 0
U 1589, 0484,0E7D,0034,0047,0151,A              NEXT/ED.RESV.OPER                ;

                                                ;10------------------------------;
                                                M[TEMP5]_ZLITO[0],               ; RETURN A ZERO
U 158A, 0D86,5C37,0030,0047,015F,E              NEXT/ED.READ.50                  ;
                                        =
```

```
                                ;34365 =10
                                ;34366 ED.READ.10:
                                ;34367          ;10----------------------------;
                                ;34368          VA_R[R1]_RB+CONX(1),          ;
U 1526, 0C84,073D,0004,44A7,014E,3 ;34369       NEXT/ED.READ.FETCH.BYTE       ;
                                ;34370
                                ;34371          ;11----------------------------;
                                ;34372          R[TEMP5]_M[MDR],              ;
U 1527, 0085,2592,4B31,58E7,0156,2 ;34373       IP.TS?,NEXT/ED.READ.DONE      ;
                                ;34374
                                ;34375 -0**
                                ;34376 ED.READ.FETCH.BYTE:
                                ;34377          ;0**---------------------------;
                                ;34378          READ,SIZE[BYTE],              ;
                                ;34379          PL_[4],                       ; USED TO FETCH LEFT NIBBLE
U 14E3, 0980,0EF6,4000,2050,015F,6 ;34380       NEXT/ED.READ.FETCH.OK         ;
                                ;34381
                                ;34382          ;1**---------------------------; CATCH RETURN + 4 FROM UTRAPPED
U 14E7, 0880,0036,40B0,0047,0000,4 ;34383       RETURN [+4]                   ; READ AT .-4, TAKE FPD RETURN.
                                ;34384
                                ;34385 ED.READ.FETCH.OK:
                                ;34386          ;---------------------------;
                                ;34387          R[TEMP5]_M[MDR].RR.P,         ;
U 15F6, 0485,2177,0B31,58E7,0156,2 ;34388       IP.TS?                        ;
                                ;34389
                                ;34390 =010
                                ;34391 ED.READ.DONE:
                                ;34392          ;010---------------------------;
                                ;34393          M[TEMP5]_MB.AND.ZLIT0[0F],    ;
U 1562, 0986,5C12,00B0,7847,0000,1 ;34394       RETURN [+1]                   ;
                                ;34395
                                ;34396          ;011---------------------------;
                                ;34397          INTPEND OR TIMER?,            ;
U 1563, 0480,0036,4AF0,0047,04F7,0 ;34398       PUSH,NEXT/IE.SERV.IP.TS2      ; WILL RETURN-1 IF ONLY TIMER
                                ;34399
                                ;34400 =111     ;111---------------------------; CATCH FPD RETURN FROM IE.SERV.IP.TS
U 1567, 0880,0036,40B0,0047,0000,4 ;34401       RETURN [+4]                   ; TAKE SPECIAL RETURN TO CALLING ROUTINE
                                ;34402
                                ;34403 ED.READ.50:
                                ;34404          ;---------------------------;
U 15FE, 0936,AD11,0030,0847,0160,2 ;34405       M[TEMP10]_MB+ZLIT16[1]        ;
                                ;34406
                                ;34407          ;---------------------------;
                                ;34408          R[R0]_M[TEMP10],              ;
U 1602, 0C84,A592,40B4,0047,0000,1 ;34409       RETURN [+1]                   ;
```

```
                                ;34410 .TOC "  Edit Packed to Character string : STORE STRING OF CHARACTERS"
                                ;34411
                                ;34412 ;*****************************************************************************
                                ;34413 ;       Entry points    ED.STORE
                                ;34414 ;
                                ;34415 ;       Input           TEMP5           Character to be stored
                                ;34416 ;                       R5              Destination address
                                ;34417 ;
                                ;34418 ;       Resources       VA              Address to be written
                                ;34419 ;
                                ;34420 ;       Output          TEMP1           TEMP5<8:0>'TEMP1<8:0> if interrupted
                                ;34421 ;                       R5              Incremented destination address
                                ;34422 ;
                                ;34423 ;       This routine writes the character in TEMP5 to the memory location
                                ;34424 ;       addressed by R5 and CHECKS FOR INTERRUPTS. When complete control is
                                ;34425 ;       returned to the calling routine.
                                ;34426 ;
                                ;34427 ;       Charlie's flows 6.12
                                ;34428 ;*****************************************************************************
                                ;34429
                                ;34430 ED.STORE:
                                ;34431          ;-------------------------------;
U 1606, 0880,05BE,4035,44A7,014F,2 ;34432       VA_R[R5]                        ;
                                ;34433
                                ;34434 =0**     ;0*-----------------------------;
                                ;34435          WRITE M[TEMP5],SIZE[BYTE],      ;
                                ;34436          MM.ALLOW.INT?,                  ;
U 14F2, 0480,5592,42C0,05D8,0157,0 ;34437       NEXT/ED.STORE.DONE              ;
                                ;34438
                                ;34439          ;1**----------------------------; RETURN + 4 FROM UTRAPPED WRITE
                                ;34440          M[TEMP5]_MB.AND.ZLITO[OFF],     ; CLEAR ALL BUT BYTE TO BE SAVED
U 14F6, 0D86,5C12,0037,F847,0157,7 ;34441       NEXT/ED.STORE.PACK              ;
                                ;34442
                                ;34443 =000
                                ;34444 ED.STORE.DONE:
                                ;34445          ;000----------------------------;
                                ;34446          VA_R[R5]_RB+CONX(1),            ;
U 1570, 0C84,073D,0085,44A7,0000,1 ;34447       RETURN [+1]                     ;
                                ;34448
                                ;34449          ;001----------------------------; CATCH RETURN-1 FORM IE.SERV.IP.TS
                                ;34450          VA_R[R5]_RB+CONX(1),            ; CONTINUE WHERE INTERRUPT OCCURED.
U 1571, 0C84,073D,0085,44A7,0000,1 ;34451       RETURN [+1]                     ;
                                ;34452
                                ;34453          ;010----------------------------;
                                ;34454          INTPEND OR TIMER?,              ;
U 1572, 0480,0036,4AF0,0047,04F7,0 ;34455       PUSH,NEXT/IE.SERV.IP.TS2        ;
                                ;34456
                                ;34457 =110     ;110----------------------------; CATCH FPD RETURN FROM IE.SERV.IP.TS
U 1576, 0D86,5C12,0037,F847,0157,7 ;34458       M[TEMP5]_MB.AND.ZLITO[OFF]      ; CLEAR ALL BUT BYTE TO BE SAVED
                                ;34459
                                ;34460 ED.STORE.PACK:
                                ;34461          ;-------------------------------;
                                ;34462          M[TEMP1]_MB.OR.(R[TEMP5].RL.8), ; SAVE CHARACTER THAT WAS BEING WRITTEN
U 1577, 0886,1292,40A1,4047,0000,2 ;34463       RETURN [+2]                     ; TAKE SPECIAL RETURN TO CALLING ROUTINE
                                ;34464 =
```

```
                        ;34465 .TOC "  Edit Packed to Character string : STORE SINGLE CHARACTER"
                        ;34466
                        ;34467 ;*****************************************************************************
                        ;34468 ;       Entry points    ED.STORE.NOINT
                        ;34469 ;
                        ;34470 ;       Input           TEMP5           Character to be stored
                        ;34471 ;                       R5              Destination address
                        ;34472 ;
                        ;34473 ;       Resources       VA              Address to be written
                        ;34474 ;
                        ;34475 ;       Output          TEMP1           TEMP5<8:0>'TEMP1<8:0> if interrupted
                        ;34476 ;                       R5              Incremented destination address
                        ;34477 ;
                        ;34478 ;       Subroutines     ED.LOOP
                        ;34479 ;                       ED.PACK
                        ;34480 ;
                        ;34481 ;       This routine writes the character in TEMP5 to the memory location
                        ;34482 ;       addressed by R5.  When complete control is transfered to ED.LOOP.
                        ;34483 ;
                        ;34484 ;       Charlie's flows 6.12
                        ;34485 ;*****************************************************************************
                        ;34486
                        ;34487 ED.STORE.NOINT:
                        ;34488          ;-------------------------------;
U 1608, 0080,05BE,4035,44A7,014F,A      ;34489          VA_R[R5]                        ;
                        ;34490
                        ;34491 =0**
                        ;34492 ED.STORE.NOINT.10:
                        ;34493          ;0**----------------------------;
                        ;34494          WRITE M[TEMP5],SIZE[BYTE],      ;
U 14FA, 0880,5592,4000,05D8,0161,4      ;34495          NEXT/ED.STORE.NOINT.DONE        ;
                        ;34496
                        ;34497          ;1**----------------------------; RETURN + 4 FROM UTRAPPED
U 14FE, 0D86,5C12,0037,F847,0160,A      ;34498          M[TEMP5]_MB.AND.ZLIT0[OFF]      ; WRITE AT .-4, TAKE FPD RETURN
                        ;34499
                        ;34500          ;-------------------------------;
U 160A, 0486,1292,4021,4047,0161,3      ;34501          M[TEMP1]_MB.OR.(R[TEMP5].RL.8)  ; SAVE CHARACTER BEING STORED
                        ;34502
                        ;34503          ;-------------------------------;
                        ;34504          M[TEMP3]_ZLIT8[04],             ;
U 1613, 0D86,3DB7,0030,2047,0165,0      ;34505          NEXT/ED.PACK                    ;
                        ;34506
                        ;34507 ED.STORE.NOINT.DONE:
                        ;34508          ;-------------------------------;
                        ;34509          VA_R[R5]_RB+CONX(1),            ;
U 1614, 0084,073D,0005,44A7,014E,2      ;34510          NEXT/ED.LOOP                    ;
```

```
                                ;34511 .TOC "  Edit Packed to Character string : EXIT"
                                ;34512
                                ;34513 ;*****************************************************************************
                                ;34514 ;       Entry points    ED.EXIT
                                ;34515 ;
                                ;34516 ;       Input           TEMP1           Original source length
                                ;34517 ;                       TEMP2           PC of next instruction
                                ;34518 ;                       MTEMP8          Current PSL
                                ;34519 ;                       R0              Remaining source length (should be 0)
                                ;34520 ;                       R1              Address of EO$END pattern operator + 1
                                ;34521 ;
                                ;34522 ;       Output          R0              Original source length
                                ;34523 ;                       R1              Address of most significant source digit
                                ;34524 ;                       R2              0
                                ;34525 ;                       R3              Address of EO$END pattern operator
                                ;34526 ;                       R4              0
                                ;34527 ;
                                ;34528 ;       This routine does the final cleanup and setting of GPR's.
                                ;34529 ;
                                ;34530 ;       Charlie's flows 6.16
                                ;34531 ;*****************************************************************************
                                ;34532
                                ;34533 ED.EXIT:
                                ;34534          ;-------------------------------;
                                ;34535          WB_R[R0],                       ; CHECK IF ALL SOURCE DIGITS WERE USED
U 1633, 0080,05BE,4A34,0047,0957,8 322*         ;34536          WX.EQ.0?                        ;
                                ;34537
                                ;34538 =0       ;0-------------------------------;
U 1578, 0480,0036,4030,0047,0151,A              ;34539          NEXT/ED.RESV.OPER               ;
                                ;34540
                                ;34541          ;1-------------------------------;
U 1579, 0C84,1592,4034,0047,0163,6              ;34542          R[R0]_M[TEMP1]                  ;
```

```
                                ;34543          ;--------------------------------;
U 1636, 0484,05B7,0034,8047,0163,B  ;34544      R[R2]_0                          ;
                                ;34545          ;--------------------------------;
                                ;34546
U 163B, 0C86,1001,803D,8047,0164,2  ;34547      M[TEMP1]_MB.SR.1                 ;
                                ;34548          ;--------------------------------;
                                ;34549
U 1642, 0084,1003,0034,4047,0164,6  ;34550      R[R1]_RB-M[TEMP1]                ;
                                ;34551          ;--------------------------------;
                                ;34552
U 1646, 0485,AE51,0034,C047,0164,A  ;34553      R[R3]_M[PC]-1                    ;
                                ;34554          ;--------------------------------;
                                ;34555          ;--------------------------------;
                                ;34556          PC_M[TEMP2],                     ;
U 164A, 04E0,2002,403D,8487,0164,E  ;34557      CLEAR FPD                        ;
                                ;34558          ;--------------------------------;
                                ;34559          ;--------------------------------;
                                ;34560          M[TEMP8]_MB.ANDNOT.ZLIT0[8],     ; PREPARE TO CLEAR PSL<N> IF NEEDED
U 164E, 0186,8C13,04B0,4047,0150,A  ;34561      FLAG2?                           ; BRANCH ON PSL<Z>
                                ;34562 =0**
                                ;34563 ED.EXIT.10:
                                ;34564          ;0**-----------------------------;
                                ;34565          R[R4]_0,                         ;
U 150A, 0084,05B7,0135,0047,003F,9  ;34566      IRD1                             ;
                                ;34567          ;--------------------------------;
                                ;34568          ;1**-----------------------------;
                                ;34569          PSL_M[TEMP8].ANDNOT.ZLIT24[8],   ; CLEAR PSL<FPD,N>
U 150E, 0D80,8C93,8030,4007,0150,A  ;34570      NEXT/ED.EXIT.10                  ;
```

```
                                ;34571 .TOC "  Edit Packed to Character string : FPD PACK ROUTINE"
                                ;34572
                                ;34573 ;*****************************************************************************
                                ;34574 ;       Entry points    ED.PACK
                                ;34575 ;                       ED.RESV.OPER
                                ;34576 ;                       ED.RESV.OPER.ABORT
                                ;34577 ;
                                ;34578 ;       Input           TEMP1           ?<31:16>'TEMP5<7:0>'SRCLEN
                                ;34579 ;                       TEMP3           0<31:16>'FPD CODE'LOOP COUNT
                                ;34580 ;
                                ;34581 ;       Output          R0              -zeros<15:0> ' (source length remaining)
                                ;34582 ;                       R1              Current source address
                                ;34583 ;                       R2              MDR<8:0>'DELTA PC'SIGN'FILL
                                ;34584 ;                       R3              Address of pattern operator causing
                                ;34585 ;                                       fault or next operator if interrupted.
                                ;34586 ;                       R4              TEMP5<7:0>'SRCLEN'FPD CODE'LOOP COUNT
                                ;34587 ;                       R5              Address of next destination byte.
                                ;34588 ;
                                ;34589 ;       Charlie's flows 6.13
                                ;34590 ;*****************************************************************************
                                ;34591
                                ;34592 ED.PACK:
                                ;34593          ;-------------------------------;
                                ;34594          R[R3]_M[PC],                    ;
U 1650, 0C85,A592,4034,C047,0151,E ;34595       NEXT/ED.PACK.10                 ;
                                ;34596 =0**
                                ;34597 ED.RESV.OPER:
                                ;34598          ;0**----------------------------; RESERVED OPERAND FAULT
                                ;34599          R[R3]_M[PC]-1,                  ;
U 151A, 0485,AE51,0034,C047,04FF,8 ;34600       PUSH,NEXT/IE.OPER.FAULT         ;
                                ;34601
                                ;34602 ED.PACK.10:
                                ;34603          ;1**----------------------------; RETURN FROM RESV OPER FAULT
U 151E, 0485,9003,0030,8047,0165,2 ;34604       R[TEMP2]_RB-M[PCBACK]           ;
```

```
                                ;34605 ED.PACK.20:
                                ;34606          ;-------------------------------;
U 1652, 0084,23BE,4014,8047,0165,4 ;34607       R[R2]_RB.OR.(M[TEMP2].RL.16)    ;
                                ;34608
                                ;34609          ;-------------------------------;
U 1654, 0D81,2C12,003",FC67,0165,6 ;34610       MDR_M[MDR].AND.ZLIT0[OFF]       ; PREPARE TO SAVE LAST SOURCE BYTE
                                ;34611
                                ;34612          ;-------------------------------;
U 1656, 0085,23BE,4004,8047,0165,B ;34613       R[R2]_RB.OR.(M[MDR].RL.24)      ; SAVE LAST SOURCE BYTE READ
                                ;34614
                                ;34615          ;-------------------------------;
U 165B, 0481,9001,0030,8487,0166,3 ;34616       PC_M[PCBACK]+R[TEMP2]           ;
                                ;34617
                                ;34618          ;-------------------------------;
                                ;34619          R[R4]_ZEXT(M[TEMP1]),           ;
U 1663, 0484,159E,4015,0047,0166,7 ;34620       SIZE[WORD]                      ; CLEAR LEFT HALF TO 'OR' IN TEMP3
                                ;34621
                                ;34622          ;-------------------------------;
                                ;34623          R[R4]_M[TEMP3].OR.(RB.RL.16),   ;
U 1667, 0484,3292,4015,0047,00FE,2 ;34624       NEXT/TE.PACK.DONE               ;
```

```
                        ;34625 .TOC "  Edit Packed to Character string : FPD UNPACK ROUTINE"
                        ;34626
                        ;34627 ;*****************************************************************************
                        ;34628 ;       Entry points    ED.UNPACK
                        ;34629 ;
                        ;34630 ;       Input           R0              -zeros<15:0> ' (source length remaining)
                        ;34631 ;                       R1              Current source address
                        ;34632 ;                       R2              MDR<8:0>'DELTA PC'SIGN'FILL
                        ;34633 ;                       R3              Address of pattern operator causing
                        ;34634 ;                                       fault or next operator if interrupted.
                        ;34635 ;                       R4              TEMP5<7:0>'SRCLEN'FPD CODE'LOOP COUNT
                        ;34636 ;                       R5              Address of next destination byte.
                        ;34637 ;
                        ;34638 ;       Output          TEMP1           0<31:8> ' (source length remaining)
                        ;34639 ;                       TEMP2           PC of next instruction
                        ;34640 ;                       TEMP3           Loop count (if appropriate)
                        ;34641 ;                       TEMP5           Character to be written (if appropriate)
                        ;34642 ;                       MTEMP8          Current PSL
                        ;34643 ;                       R2              0<31:16> ' SIGN ' FILL
                        ;34644 ;                       FPDOFFSET       0
                        ;34645 ;                       MDR             ?<31:8> ' (last source byte read)
                        ;34646 ;
                        ;34647 ;       Charlie's flows 6.14
                        ;34648 ;*****************************************************************************
                        ;34649
                        ;34650 .REGION/IRD1.R1L,IRD1.R1H
                        ;34651 =000
                        ;34652 ED.UNPACK:
                        ;34653                 ;-------------------------------;
U 0388, 0486,22B7,0014,8047,0166,8      ;34654         M[TEMP2]_R[R2].RR.16
                        ;34655 =
                        ;34656 .REGION/EDIT.R1L,EDIT.R1H/EDIT.R2L,EDIT.R2H/EDIT.R3L,EDIT.R3H
                        ;34657                 ;-------------------------------;
U 166B, 0186,2C12,0037,F847,0167,3      ;34658         M[TEMP2]_MB.AND.ZLITO[OFF]      ;
                        ;34659
                        ;34660                 ;-------------------------------;
U 1673, 0885,9001,0030,8047,0167,7      ;34661         R[TEMP2]_M[PCBACK]+RB           ;
```

```
                                                          ;---------------------------------;
U 1677, 0880,02B7,0024,8467,0167,B                        MDR_R[R2].RR.24                   ; RESTORE LAST SOURCE BYTE READ

                                                          ;---------------------------------;
U 167B, 0886,75BE,4034,8047,0167,D                        M[TEMP7]_R[R2]                    ;

                                                          ;---------------------------------;
                                                          R[R2]_ZEXT(M[TEMP7]),             ;
U 167D, 0484,759E,4014,8047,0168,3                        SIZE[WORD]                        ;

                                                          ;---------------------------------;
U 1683, 0080,05BE,4034,C487,0168,B                        PC_R[R3]                          ;

                                                          ;---------------------------------;
U 168B, 0C86,12B7,0015,0047,0168,E                        M[TEMP1]_R[R4].RR.16              ;

                                                          ;---------------------------------;
U 168E, 0986,1C12,0037,F847,0169,3                        M[TEMP1]_MB.AND.ZLITO[OFF]        ;

                                                          ;---------------------------------;
U 1693, 0D86,CC37,0030,0047,0169,7                        M[FPDOFFSET]_ZLITO[0]             ;

                                                          ;---------------------------------;
U 1697, 0086,75BE,4034,4047,0169,8                        M[TEMP7]_R[R1]                    ; PREPARE TO ADJUST SRCADDR

                                                          ;---------------------------------;
U 1698, 0880,05BE,4274,0047,0153,6                        WB_R[R0],WB<1-0>?                 ; BRANCH ON RIGHT OR LEFT NIBBLE
                                              =10
                                              ED.UNPACK.03:
                                                          ;10-------------------------------;
                                                          M[TEMP3]_R[R4],                   ;
U 1536, 0486,35BE,4035,0047,0169,E                        NEXT/ED.UNPACK.05                 ;

                                                          ;11-------------------------------;
                                                          R[TEMP7]_RB-1,                    ; WILL BE LOADED INTO R1 IF
U 1537, 0C84,0E7D,0031,C047,0153,6                        NEXT/ED.UNPACK.03                 ; INTERRUPTED DURING A READ

                                              ED.UNPACK.05:
                                                          ;---------------------------------;
U 169E, 0D86,3C12,0037,F847,016A,0                        M[TEMP3]_MB.AND.ZLITO[OFF]        ;

                                                          ;---------------------------------;
U 16A0, 0086,8036,4030,0087,016A,2                        M[TEMP8]_PSL                      ;

                                                          ;---------------------------------;
U 16A2, 0980,8C12,0030,7807,016A,E                        FLAGS_M[TEMP8].AND.ZLITO[OF]      ;

                                                          ;---------------------------------;
U 16AE, 0486,C2B7,0005,0047,016B,3                        M[TEMP0]_R[R4].RR.8               ;

                                                          ;---------------------------------;
                                                          WB_M[TEMP0]-ZLITO[0B],            ; CHECK FOR VALID PACK CODE.
U 16B3, 0980,0C10,0840,5847,0955,5 413*                   SIZE[BYTE],SIGND CMP DEF?         ;
```

```
                                ;34715 =01      ;01------------------------------;
U 1555, 0C80,0036,4030,0047,003B,0              ;34716          NEXT/IE.OPCOD.DEC               ; SOMEBODY MESSED UP THE REGISTERS.
                                ;34717
                                ;34718          ;11------------------------------;
                                ;34719          WB_R[R4].RR.8,                  ;
U 1557, 0880,02B7,0205,0047,095B,0 337*         ;34720          WB<5-0>?                        ;
                                ;34721
                                ;34722 =110000
                                ;34723          ;110000--------------------------;
                                ;34724          R[R1]_M[TEMP7],                 ; ADJUST SRCADDR TO REDO READ
U 15B0, 0084,7592,4034,4047,0153,1              ;34725          NEXT/ED.MOVE.P1                 ;
                                ;34726
                                ;34727          ;110001--------------------------;
                                ;34728          R[R1]_M[TEMP7],                 ; ADJUST SRCADDR TO REDO READ
U 15B1, 0884,7592,4034,4047,0154,1              ;34729          NEXT/ED.FLOAT.P1                ;
                                ;34730
                                ;34731          ;110010--------------------------;
                                ;34732          R[R1]_M[TEMP7],                 ; ADJUST SRCADDR TO REDO READ
U 15B2, 0884,7592,4034,4047,0152,1              ;34733          NEXT/ED.ADJUST.P1               ;
                                ;34734
                                ;34735          ;110011--------------------------;
                                ;34736          M[TEMP5]_R[R4].RR.24,           ; RESTORE CHARACTER TO BE WRITTEN
U 15B3, 0086,52B7,0025,0047,0153,2              ;34737          NEXT/ED.MOVE.P2                 ;
                                ;34738
                                ;34739          ;110100--------------------------;
                                ;34740          M[TEMP5]_R[R4].RR.24,           ; RESTORE CHARACTER TO BE WRITTEN
U 15B4, 0086,52B7,0025,0047,0160,8              ;34741          NEXT/ED.STORE.NOINT             ;
                                ;34742
                                ;34743          ;110101--------------------------;
                                ;34744          M[TEMP5]_R[R4].RR.24,           ; RESTORE CHARACTER TO BE WRITTEN
U 15B5, 0886,52B7,0025,0047,0154,2              ;34745          NEXT/ED.FLOAT.P2                ;
                                ;34746
                                ;34747          ;110110--------------------------;
U 15B6, 0C80,0036,4030,0047,0155,9              ;34748          NEXT/ED.FILL                    ;
                                ;34749
                                ;34750          ;110111--------------------------;
U 15B7, 0C80,0036,4030,0047,0156,9              ;34751          NEXT/ED.FILL.P2                 ;
                                ;34752
                                ;34753          ;111000--------------------------;
                                ;34754          M[TEMP6]_R[TEMP3],              ;
U 15B8, 0C86,65BE,4030,C047,0151,4              ;34755          NEXT/ED.BLANK.ZERO.LOOP         ;
                                ;34756
                                ;34757          ;111001--------------------------;
                                ;34758          M[TEMP0]_D_R[TEMP3],            ; RESTORE PATTERN OPERATOR
U 15B9, 0086,05BE,6030,C047,014F,3              ;34759          NEXT/ED.POD.15                  ;
                                ;34760
                                ;34761          ;111010--------------------------;
U 15BA, 0480,0036,4030,0047,014E,2              ;34762          NEXT/ED.LOOP                    ;
                                ;34763 =
```

```
                                        ;34764 .TOC "  Edit Packed to Character string : Ird & Dsize Rom Definition"
                                        ;34765
;34766  .NOBIN
;34767
;34768
;34769
;34770
;34771  .ICODE  ;       OPS             REG                     MEM             OPS         FPA REG                 FPA MEM
;34772       038: FPD [NOP][ED.UNPACK            ]                           [NOP][ED.UNPACK             ],                  ;EDITPC
;34773            IRD1[LOD][OS.RED               ]                           [LOD][OS.RED                ]
;34774  .OCODE
;34775       038: CNTO[NOP][ED.EDITPC            ][ED.EDITPC             ]  [NOP][ED.EDITPC             ][ED.EDITPC             ],
;34776            CNT1[NOP][ED.EDITPC.02         ][ED.EDITPC.02          ]  [NOP][ED.EDITPC.02          ][ED.EDITPC.02          ]
;34777
;34778
;34779
;34780
;34781  .DCODE
;34782  038:    SIZE [WORD] [BYTE] [BYTE] [BYTE] [   0] [   0]  ;EDITPC
;34783
;34784  .UCODE
                                        ;34785 .BIN
```

34785; This page intentionally left blank.

```
                                        ;34786 .TOC "MPRCHM.MIC"
                                        ;34787 .TOC "REVISION 38.0"
                                        ;34788 ;       Jeff Peng, Brian Allison, CHARLIE MCDOWELL
                                        ;34789
;34790  .NOBIN
;34791
;34792  .TOC "  Revision History"
;34793
;34794  ; REV   EXPLAINATION
;34795
;34796  ; 38    Jump to PCS to pick up microrev number
;34797  ; 37    Add new IPR number 63(decimal) to check for TB hits.
;34798  ;       Change MTPR TBDR to take a reserved operand fault if attempting
;34799  ;       to disable both halves of the TB.
;34800  ;       Assign permanent addresses to free locations (where possible)
;34801  ; 36    Add a BUS/PRB.RD.PTE.K to microword 0DFA to cure a deadlock problem
;34802  ;       that occurs when running 3 massbuses at the same time.
;34803  ; 35    Fix deposit to IORESET from the console to not take interrupts.
;34804  ;       Have halts detected in CHM tell console to look at the front panel.
;34805  ;       Change MFPR ACCS to use branch on FPA present not IRD rom.
;34806  ; 34    Initial release.
                                        ;34807          .BIN
```

```
                        ;34808 .TOC     "       MTPR, MFPR, CHMX                : SELECT IPR"
                        ;34809
                        ;34810 ;*****************************************************************************
                        ;34811 ;       Entry points    MP.SELECT.IPR
                        ;34812 ;
                        ;34813 ;       Input           TEMP1           Internal Processor Register number
                        ;34814 ;
                        ;34815 ;       Output          TEMP2           IPR(TEMP1) if TEMP1 is 0-D
                        ;34816 ;
                        ;34817 ;       Resources       RNUM
                        ;34818 ;                       FLAG3           Cleared if priviledged instruction
                        ;34819 ;
                        ;34820 ;       Subroutines     IE.OPCOD.DEC
                        ;34821 ;
                        ;34822 ;       THIS ROUTINE DECODES THE REGISTER NUMBER IN MTPR AND MFPR
                        ;34823 ;       INSTRUCTIONS.
                        ;34824 ;       IT DOES A RETURN + 1 IF IPR(RNUM) IS THE CORRECT REGISTER.
                        ;34825 ;       IT DOES A RETURN + 2 IF SP IS THE CORRECT REGISTER.
                        ;34826 ;       IT DOES A RETURN + 3 IF IT IS A LEGAL REGISTER NOT ONE OF THE ABOVE.
                        ;34827 ;       IT DOES A RETURN + 4 IF AN ILLEGAL REGISTER NUMBER.
                        ;34828 ;       IT TAKES A PRIVILEDGED INSTRUCTION FAULT IF THE MODE IS NOT ZERO.
                        ;34829 ;
                        ;34830 ;       CHARLIE'S FLOWS 3.2
                        ;34831 ;*****************************************************************************
                        ;34832
                        ;34833 .REGION/MPRCHM.R1L,MPRCHM.R1H/MPRCHM.R2L,MPRCHM.R2H/MPRCHM.R3L,MPRCHM.R3H
                        ;34834 =1100
                        ;34835 MP.SELECT.IPR:
                        ;34836         ;1100-------------------------;
                        ;34837         WB_M[TEMP1].ANDNOT.ZLITO[OF],   ; SEPERATE REGISTER NUMBERS 0-F FROM
U 0DAC, 0980,1C13,8A30,7847,00DB,8      ;34838         WX.EQ.0?,NEXT/MP.SELECT.IPR10   ; THE REST AND HANDLE SEPERATELY.
                        ;34839
                        ;34840         ;1101-------------------------;
                        ;34841         WB_M[FPDOFFSET],                ;
U 0DAD, 0880,C592,46F0,0047,00DA,9      ;34842         WB<31-30>?,NEXT/MP.SELECT.PI    ;
                        ;34843
                        ;34844         ;1110-------------------------;
                        ;34845         WB_M[FPDOFFSET],                ;
U 0DAE, 0880,C592,46F0,0047,00DA,9      ;34846         WB<31-30>?,NEXT/MP.SELECT.PI    ;
                        ;34847
                        ;34848         ;1111-------------------------;
                        ;34849         WB_M[FPDOFFSET],                ;
U 0DAF, 0880,C592,46F0,0047,00DA,9      ;34850         WB<31-30>?,NEXT/MP.SELECT.PI    ;
                        ;34851
                        ;34852 =01
                        ;34853 MP.SELECT.PI:
                        ;34854         ;0--------------------------; NOT FROM CONSOLE MODE
                        ;34855         CLEAR FLAG3,                    ; PUSH PCBACK-2
U 0DA9, 0418,0036,4030,0047,003B,0      ;34856         NEXT/IE.OPCOD.DEC               ; PRIVILEDGED INSTRUCTION
                        ;34857
                        ;34858         ;1--------------------------; FROM CONSOLE MODE
U 0DAB, 0C80,0036,4030,0047,00DA,C      ;34859         NEXT/MP.SELECT.IPR              ; IGNORE PSL<CURM>.
```

```
                                ;34860 =0
                                ;34861 MP.SELECT.IPR10:
                                ;34862          ;0--------------------------------;
                                ;34863          WB_M[TEMP1],ANDNOT,ZLIT0[3F],   ; REGISTER NUMBER MUST BE LESS THAN 64.
U 0DB8, 0180,1C13,8A31,F847,00DB,C ;34864       WX.EQ.0?,NEXT/MP.SELECT.IPR50   ;
                                ;34865
                                ;34866          ;1--------------------------------;
U 0DB9, 0881,D5BE,4BB0,4047,00DC,0 ;34867       RNUM_R[TEMP1] IPR CHECK?        ; DECODE REGISTER NUMBERS IN 0-F RANGE.
                                ;34868
                                ;34869 =00
                                ;34870 MP.SELECT.IPR20:
                                ;34871          ;00-------------------------------; REGISTER NUMBER = 8,9,10,11,12, OR 13.
                                ;34872          M[TEMP2]_R[IPR.R],              ; READ IPR(RNUM) FOR MFPR.
U 0DC0, 0886,25BE,40BC,8047,0000,1 ;34873       RETURN [+1]                     ; SELECT IPR(RNUM).
                                ;34874
                                ;34875          ;01-------------------------------;
                                ;34876          M[TEMP2]_R[SP],                 ; READ SP FOR MFPR.
U 0DC1, 0C86,25BE,40B7,8047,0000,2 ;34877       RETURN [+2]                     ; SELECT SP.
                                ;34878
                                ;34879          ;10-------------------------------; REGISTER NUMBER = 5,6,7,14, OR 15.
U 0DC2, 0880,0036,40B0,0047,0000,4 ;34880       RETURN [+4]                     ; ILLEGAL REGISTER NUMBER.
                                ;34881
                                ;34882          ;11-------------------------------; REGISTER NUMBER = 0,1,2,3, OR 4.
U 0DC3, 0480,0036,47B0,0207,08DB,B 479* ;34883  PSL<IS.CURM>?                    ; CHECK IF IN KERNAL OR INTERRUPT MODE.
                                ;34884
                                ;34885 =1011    ;1011-----------------------------;
                                ;34886          WB_M[TEMP1],                    ; IF SELECTING KSP WHILE IN KERNAL MODE
U 0DBB, 0480,1592,4A30,0047,08DC,0 322* ;34887  WX.EQ.0?,NEXT/MP.SELECT.IPR20    ; SELECT SP INSTEAD.
                                ;34888
                                ;34889          ;1111-----------------------------;
                                ;34890          WB_M[TEMP1]-ZLIT0[4],           ; IF SELECTING ISP WHILE ON INTERRUPT
U 0DBF, 0180,1C10,0A30,2047,08DC,0 384* ;34891  WX.EQ.0?,NEXT/MP.SELECT.IPR20    ; STACK, SELECT SP INSTEAD.
                                ;34892
                                ;34893 =0
                                ;34894 MP.SELECT.IPR50:
                                ;34895          ;0--------------------------------; REGISTER NUMBER IS GREATER THAN 63.
U 0DBC, 0880,0036,40B0,0047,0000,4 ;34896       RETURN [+4]                     ; ILLEGAL REGISTER NUMBER.
                                ;34897
                                ;34898          ;1--------------------------------; REGISTER NUMBER IS IN RANGE 16.-63.
                                ;34899          WB_M[TEMP1]-ZLIT0[41.],         ; CHECK IN IF RANGE 41.-54.
U 0DBD, 0580,1C10,06F1,4847,08DE,9 377* ;34900  WB<31-30>?                       ;
                                ;34901
                                ;34902 =01      ;01-------------------------------; REGISTER NUMBER IS GREATER THAN 40.
                                ;34903          WB_M[TEMP1]-ZLIT0[55.],         ; CHECK IF LESS THAN 55.
U 0DE9, 0180,1C10,06F1,B847,08DE,D 377* ;34904  WB<31-30>?,NEXT/MP.SELECT.IPR60 ;
                                ;34905
                                ;34906          ;11-------------------------------; REGISTER NUMBER IS IN RANGE 16.-40.
                                ;34907          PL_[24.]                        ; LOAD PL FOR USE BY SOME MXPR'S
U 0DEB, 0D80,0EF6,40B0,C047,0000,3 ;34908       RETURN [+3]                     ; RETURN AND DO A WIDE BRANCH ON TEMP1.
```

```
                                ;34909 =01
                                ;34910 MP.SELECT.IPR60:
                                ;34911          ;01-----------------------------; REGISTER NUMBER IS IN RANGE 55.-63.
                                ;34912          PL_[24.],                       ; LOAD PL FOR USE BY SOME MXPR'S
U 0DED, 0D80,0EF6,40B0,C047,0000,3  ;34913      RETURN [+3]                     ; RETURN AND DO A WIDE BRANCH ON TEMP1.
                                ;34914
                                ;34915          ;11-----------------------------; REGISTER NUMBER IS IN RANGE 41.-54.
U 0DEF, 0880,0036,40B0,0047,0000,4  ;34916      RETURN [+4]                     ; ILLEGAL REGISTER NUMBER.
```

```
                        ;34917 .TOC "  MTPR, MFPR, CHMX                    : MTPR"
                        ;34918
                        ;34919 ;*************************************************************************************
                        ;34920 ;       MTPR    src.rl, procreg.rl
                        ;34921 ;
                        ;34922 ;       Input           Q               Source operand
                        ;34923 ;                       MDR             Destination processor register number
                        ;34924 ;
                        ;34925 ;       Resources       MM.TEMP0        Temp for writing interval timer regs
                        ;34926 ;                       TEMP1           Pass register # to SELECT.IPR, and temp
                        ;34927 ;                       TEMP2           Temporary
                        ;34928 ;                       MSCAR
                        ;34929 ;                       STEPC
                        ;34930 ;                       D               Temporary
                        ;34931 ;                       LONLIT          MBZ checks
                        ;34932 ;                       PL              Getting MSB set in rotater
                        ;34933 ;                       VA
                        ;34934 ;
                        ;34935 ;       Subroutines     SELECT.IPR
                        ;34936 ;                       IE.OPER.FAULT
                        ;34937 ;                       IE.ICR.SERV2
                        ;34938 ;                       IE.SERV.IP.TS2
                        ;34939 ;
                        ;34940 ;       MTPR IS BROKEN INTO 3 SECTIONS.
                        ;34941 ;       FOR GROUP 1 THE REGISTER NUMBER CORRESPONDS WITH RNUM FOR IPR(RNUM).
                        ;34942 ;       FOR GROUP 2 SP (THE CURRENT STACK POINTER) SHOULD BE USED
                        ;34943 ;               INSTEAD OF KSP OR ISP.
                        ;34944 ;       FOR GROUP 3 A WIDE BRANCH MUST BE DONE ON THE REGISTER NUMBER.
                        ;34945 ;       CHARLIE'S FLOWS 3.2
                        ;34946 ;*************************************************************************************
                        ;34947
                        ;34948 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;34949 =000
                        ;34950 MP.MTPR:
                        ;34951                 ;000-------------------------------;
                        ;34952                 R[TEMP1]_M[MDR],PSL<IS.CURM>?,  ; DECODE THE REGISTER NUMBER INTO
U 0390, 0885,2592,4780,4207,0CDA,C 479*         ;34953                 PUSH,NEXT/MP.SELECT.IPR         ; ONE OF THREE GROUPS.
                        ;34954
                        ;34955                 ;001-------------------------------;                      **MBZ CHECK**
                        ;34956                 RNUM_R[TEMP1] IPR CHECK?,       ; SEPERATE RNUM 0-4 FROM OTHERS.
U 0391, 0081,D5BE,4BB0,4047,00DB,6              ;34957                 NEXT/MP.MTPR.GRP1               ;
                        ;34958
                        ;34959                 ;010-------------------------------;
                        ;34960                 R[SP]_Q_Q_D,                    ; GROUP 2, USE SP.
U 0392, 0C84,002C,7037,8047,00E7,8              ;34961                 NEXT/MP.MTPR.CHK.CON            ;
                        ;34962
                        ;34963                 ;011-------------------------------;
U 0393, 0086,103A,403D,8047,00D9,D              ;34964                 M[TEMP1]_Q,NEXT/MP.MTPR.SPLIT   ; GROUP 3, MOVE SOURCE TO TEMP1
                        ;34965
                        ;34966                 ;111-------------------------------;
                        ;34967                 WB_M[FPDOFFSET],                ;
U 0394, 0080,C592,46F0,0047,00E5,1              ;34968                 WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ; BAD IPR NUMBER FORM MP.SELECT.IPR
                        ;34969 =
```

```
                                ;34970 .REGION/MPRCHM.R1L,MPRCHM.R1H/MPRCHM.R2L,MPRCHM.R2H/MPRCHM.R3L,MPRCHM.R3H
                                ;34971 =10
                                ;34972 MP.MTPR.GRP1:
                                ;34973          ;10-----------------------------; DO MBZ CHECKS ON NUMBERS 8,9,A,B,C,D.
U 0DB6, 0086,103A,403D,8047,00D9,D ;34974       M[TEMP1]_Q,NEXT/MP.MTPR.SPLIT   ; MOVE SOURCE TO TEMP1
                                ;34975
                                ;34976          ;11-----------------------------;                          **MBZ CHECK**
                                ;34977          R[IPR.R]_Q_Q_D,                 ;
U 0DB7, 0884,002C,703C,8047,00E7,8 ;34978       NEXT/MP.MTPR.CHK.CON            ;
                                ;34979
                                ;34980 MP.MTPR.SPLIT:
                                ;34981          ;-----------------------------;
                                ;34982          WB_M[MDR],                      ; GROUP 3, DO A WIDE BRANCH.
                                ;34983          WB<5-0>?,                       ;
U 0D9D, 0C81,2592,4230,0047,00DC,0 ;34984       NEXT/MP.MTPR.BRANCH             ;
                                ;34985
                                ;34986 =000000
                                ;34987 MP.MTPR.BRANCH:
                                ;34988 =001000
                                ;34989          ;001000-------------------------;                          **MBZ CHECK**
                                ;34990          R[P0BR]_Q_Q_D,                  ;
                                ;34991          WB<1-0>.NE.0?,                  ;
U 0DC8, 0084,002C,73BA,0047,00DC,4 ;34992       NEXT/MP.MTPR.P0BR               ;
                                ;34993
                                ;34994          ;001001-------------------------;                          **MBZ CHECK**
                                ;34995          LONLIT_[0F8C00000],             ;
U 0DC9, 0B80,0039,FFFF,F847,00D9,E ;34996       NEXT/MP.MTPR.PXLR               ;
                                ;34997
                                ;34998          ;001010-------------------------;                          **MBZ CHECK**
                                ;34999          R[P1BR]_D_Q_Q_D,                ;
                                ;35000          WB<1-0>.NE.0?,                  ;
U 0DCA, 0884,002D,73BA,8047,00DF,0 ;35001       NEXT/MP.MTPR.P1BR               ;
                                ;35002
                                ;35003          ;001011-------------------------;                          **MBZ CHECK**
                                ;35004          LONLIT_[7FC00000],              ;
U 0DCB, 0B80,0401,FFFF,F847,00D9,E ;35005       NEXT/MP.MTPR.PXLR               ;
                                ;35006
                                ;35007          ;001100-------------------------;                          **MBZ CHECK**
                                ;35008          R[SBR]_D_Q_Q_D,                 ;
                                ;35009          WB<1-0>.NE.0?,                  ;
U 0DCC, 0884,002D,73BB,0047,00E3,0 ;35010       NEXT/MP.MTPR.SBR                ;
                                ;35011
                                ;35012          ;001101-------------------------;                          **MBZ CHECK**
                                ;35013          PL_[22.],                       ;
U 0DCD, 0180,0EF6,4030,B047,00DA,8 ;35014       NEXT/MP.MTPR.SLR                ;
```

```
                                ;35015 =010000
                                ;35016          ;010000------------------------; REGISTER NUMBER 10 = PCBB.
                                ;35017          R[PCBB]_Q Q_D,                  ;
                                ;35018          WB<1-0>.NE.0?,                  ;
U 0DD0, 0484,002C,73B9,4047,00E6,8 ;35019       NEXT/MP.MTPR.PCBB               ;
                                ;35020
                                ;35021          ;010001------------------------; REGISTER NUMBER 11 = SCBB.
                                ;35022          LONLIT_[3FFFFE00],              ;
U 0DD1, 0380,0600,000F,F847,00DA,A ;35023       NEXT/MP.MTPR.SCBB               ;
                                ;35024
                                ;35025          ;010010------------------------; REGISTER NUMBER 12 = IPL.
                                ;35026          IPL_M[TEMP1].RL.16,             ;
U 0DD2, 0880,13B7,0010,07A7,00E7,8 ;35027       NEXT/MP.MTPR.CHK.CON            ;
                                ;35028
                                ;35029          ;010011------------------------; REGISTER NUMBER 13 = ASTLVL.
                                ;35030          M[TEMP1]_MB.AND.ZLIT0[7],       ; CLEAR MBZ BITS
U 0DD3, 0586,1C12,0030,3847,00DB,0 ;35031       NEXT/MP.MTPR.ASTLVL             ;
                                ;35032
                                ;35033          ;010100------------------------; REGISTER NUMBER  14 = SIRR.
                                ;35034          PL_MSS_M[SISR],                 ;
U 0DD4, 0080,F9C2,403D,8047,00DB,3 ;35035       NEXT/MP.MTPR.SIRR               ;
                                ;35036
                                ;35037          ;010101------------------------; REGISTER NUMBER 15 = SISR.
                                ;35038          M[SISR]_D_Q_Q_MB,               ;
U 0DD5, 0886,F00D,72B0,0047,00E0,6 ;35039       WB<0>?,NEXT/MP.MTPR.SISR05      ; CHECK MBZ BIT.
                                ;35040
                                ;35041          ;010110------------------------; REGISTER NUMBER 16 IS RESERVED.
                                ;35042          WB_M[FPDOFFSET],                ;
U 0DD6, 0080,C592,46F0,0047,00E5,1 ;35043       WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                ;35044
                                ;35045          ;010111------------------------; REGISTER NUMBER 17 = CMIERR.
                                ;35046          WB_M[FPDOFFSET],                ; CMIERR IS READ ONLY.
U 0DD7, 0080,C592,46F0,0047,00E5,1 ;35047       WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                ;35048
                                ;35049          ;011000------------------------; REGISTER NUMBER 18 = ICCS.
                                ;35050          M[MM.TEMP0]_TCSR.IICR,          ;
U 0DD8, 0886,D036,4030,01E7,00E6,D ;35051       NEXT/MP.MTPR.ICCS05             ;
                                ;35052
                                ;35053          ;011001------------------------; REGISTER NUMBER 19 = NICR.
                                ;35054          M[MM.TEMP0]_TCSR.IICR,          ;
U 0DD9, 0086,D036,4030,01E7,00E0,F ;35055       NEXT/MP.MTPR.NICR05             ;
                                ;35056
                                ;35057          ;011010------------------------; REGISTER NUMBER 1A = ICR (READ ONLY).
                                ;35058          WB_M[FPDOFFSET],                ;
U 0DDA, 0080,C592,46F0,0047,00E5,1 ;35059       WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                ;35060
                                ;35061          ;011011------------------------; REGISTER NUMBER 1B = TODR.
                                ;35062          TOYCR_M[TEMP3]_ZLIT16[57],      ; SETUP ADDRESS AND RESET TOYCN.
                                ;35063          STEPC_14.,                      ; STEPC WILL BE USED FOR LOOPING.
U 0DDB, 01B6,3D37,0032,B9A7,00E5,2 ;35064       NEXT/MP.MTPR.TODR               ;
```

```
                                        ;35065          ;011100-------------------------; REGISTER NUMBER 1C = CSRS.
                                        ;35066          TRAR_ZLIT16[80],                ; TRAR_2
U 0DDC, 0D80,0D37,0034,0267,00E6,7      ;35067          NEXT7MP.MTPR.TU58               ;
                                        ;35068
                                        ;35069          ;011101-------------------------; REGISTER NUMBER 1D = CSRD (READ ONLY).
                                        ;35070          WB_M[FPDOFFSET],                ;
U 0DDD, 0080,C592,46F0,0047,00E5,1      ;35071          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35072
                                        ;35073          ;011110-------------------------; REGISTER NUMBER 1E = CSTS.
                                        ;35074          WB_M[TEMP1],WB<0>?,             ; CHECK IF SETTING BREAK.
U 0DDE, 0480,1592,4280,0047,00E6,E      ;35075          NEXT/MP.MTPR.CSTS               ;
                                        ;35076
                                        ;35077          ;011111-------------------------; REGISTER NUMBER 1F = CSTD.
                                        ;35078          TRAR_ZLIT16[0],                 ;
U 0DDF, 0980,0D37,0030,0267,00E6,7      ;35079          NEXT7MP.MTPR.TU58               ;
                                        ;35080
                                        ;35081          ;100000-------------------------; REGISTER NUMBER 20 = RXCS.
                                        ;35082          CRAR_ZLIT16[80],                ; CRAR_2
U 0DE0, 0580,0D37,0034,0247,00E8,A      ;35083          NEXT7MP.MTPR.CONSOLE            ;
                                        ;35084
                                        ;35085          ;100001-------------------------; REGISTER NUMBER 21 = RXDB (READ ONLY).
                                        ;35086          WB_M[FPDOFFSET],                ;
U 0DE1, 0080,C592,46F0,0047,00E5,1      ;35087          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35088
                                        ;35089          ;100010-------------------------; REGISTER NUMBER 22 = TXCS.
                                        ;35090          CRAR_ZLIT16[40],                ; CRAR_1
U 0DE2, 0580,0D37,0032,0247,00E8,A      ;35091          NEXT7MP.MTPR.CONSOLE            ;
                                        ;35092
                                        ;35093          ;100011-------------------------; REGISTER NUMBER 23 = TXDB.
                                        ;35094          R[TEMP2] M[TEMP1].CLR1B,        ; CHECK IF ID FIELD IS 00.
U 0DE3, 0C84,14F7,0A30,8047,00E7,2      ;35095          WX.EQ.0?,NEXT/MP.MTPR.TXDB      ;
                                        ;35096
                                        ;35097          ;100100-------------------------; REGISTER NUMBER 24 = TBDR.
                                        ;35098          WB_.NOT.Q,WB<1-0>.NE.0?,        ; CHECK IF DISABLING BOTH HALVES.
U 0DE4, 0C80,00B8,03BD,8047,08E7,6 403* ;35099          NEXT/MP.MTPR.TBDR               ;
                                        ;35100
                                        ;35101          ;100101-------------------------; REGISTER NUMBER 25 = CADR.
                                        ;35102          WB_.NOT.Q,WB<0>?,               ; SEE IF TURNING ON CACHE.
U 0DE5, 0880,00B8,02BD,8047,08E4,0 400* ;35103          NEXT/MP.MTPR.CADR               ;
                                        ;35104
                                        ;35105          ;100110-------------------------; REGISTER NUMBER 26 = MCESR.
                                        ;35106          MEMSCAR_ZLIT24[8],              ;
U 0DE6, 0580,0CB7,0030,4687,00E8,E      ;35107          NEXT/MP.MTPR.MEMERR             ;
                                        ;35108
                                        ;35109          ;100111-------------------------; REGISTER NUMBER 27 = CAER.
                                        ;35110          MEMSCAR_ZLIT24[4],              ;
U 0DE7, 0980,0CB7,0030,26 ',0E8,E       ;35111          NEXT/MP.MTPR.MEMERR             ;
                                        ;35112
                                        ;35113          ;101000-------------------------; REGISTER NUMBER 28 = ACCS.
                                        ;35114          PL_[10.],                       ; CHECK IF ENABLING OR DISABLING.
U 0DE8, 0180,0EF6,4030,5047,00E9,0      ;35115          NEXT/MP.MTPR.ACCS               ;
```

```
                                ;35116 =110111
                                ;35117          ;110111-------------------------; REGISTER NUMBERS 29 TO 36 ARE RESERVED
                                ;35118          M[TEMP1] ZLIT0[400.],           ; REGISTER NUMBER 37 = IORESET.
U 0DF7, 0D86,1C37,003C,8047,00E7,9              ;35119          NEXT/MP.MTPR.IORESET            ;
                                ;35120
                                ;35121          ;111000-------------------------;
                                ;35122          MEMSCAR_ZLIT24[0],              ; REGISTER NUMBER 38 = MME.
U 0DF8, 0980,0CB7,0030,0687,00E9,1              ;35123          NEXT/MP.MTPR.MME                ;
                                ;35124
                                ;35125          ;111001-------------------------; REGISTER NUMBER 39 = TBIA.
                                ;35126          VA_D_ZLIT0[0],                  ; INITIALIZE D AND
U 0DF9, 0108,0C37,2030,04A7,00E4,8              ;35127          CLEAR FLAG1,NEXT/MP.MTPR.TBIA   ; CLEAR FLAG FOR CALL MP.INVALIDATE.TB
                                ;35128
                                ;35129          ;111010-------------------------; REGISTER NUMBER 3A = TBIS.
                                ;35130          VA_Q,                           ;
                                ;35131          COMPLETE CPU BUS CYCLES,        ; ADD THIS TO AVOID MASSBUS PROBLEM
U 0DFA, 0480,003A,403D,84B7,00E9,3              ;35132          NEXT/MP.MTPR.TBIS               ;
                                ;35133
                                ;35134          ;111011-------------------------; REGISTER NUMBER 3B = TBDATA.
                                ;35135          VA_R[P0BR],                     ; LOAD ADDRESS FOR WRITING TB.
U 0DFB, 0C80,05BE,403A,04A7,00E9,5              ;35136          NEXT/MP.MTPR.TBDATA             ;
                                ;35137
                                ;35138          ;111100-------------------------; REGISTER NUMBER 3C IS RESERVED.
                                ;35139          WB_M[FPDOFFSET],                ;
U 0DFC, 0080,C592,46F0,0047,00E5,1              ;35140          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                ;35141
                                ;35142          ;111101-------------------------; REGISTER NUMBER 3D = PMR.
                                ;35143          PME_Q.RR.1,                     ; WRITE HARDWARE PME.
U 0DFD, 0480,0339,803D,8287,00E7,8              ;35144          NEXT/MP.MTPR.CHK.CON            ;
                                ;35145
                                ;35146          ;111110-------------------------; REGISTER NUMBER 3E = SID (READ ONLY).
                                ;35147          WB_M[FPDOFFSET],                ;
U 0DFE, 0080,C592,46F0,0047,00E5,1              ;35148          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                ;35149
                                ;35150          ;111111-------------------------; REGISTER NUMBER 3F = TBCK
                                ;35151          WB_M[FPDOFFSET],                ;
U 0DFF, 0880,C592,46F0,0047,017F,8              ;35152          WB<31-30>?,NEXT/MP.MTPR.TBCK    ; CHECK FOR TB HIT
```

```
                                        ;35153          ;********************************* CONTINUATION OF GROUP 1 REGISTERS THAT
                                        ;35154          ;********************************* TAKE MORE THAN ONE WORD TO COMPLETE.
                                        ;35155
                                        ;35156 =0
                                        ;35157 MP.MTPR.P0BR:
                                        ;35158          ;0-------------------------------;                          **MBZ CHECK**
                                        ;35159          WB_M[FPDOFFSET],STEPC_2,        ; SETUP STEPC FOR CONSOL
U 0DC4, 00A0,C592,46F0,0047,017D,9      ;35160          WB<31-30>?,NEXT/MP.MTPR.DONE    ;
                                        ;35161
                                        ;35162          ;1-------------------------------;                          **MBZ CHECK**
                                        ;35163          WB_M[FPDOFFSET],                ;
U 0DC5, 0080,C592,46F0,0047,00E5,1      ;35164          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35165
                                        ;35166 MP.MTPR.PXLR:
                                        ;35167          ;-------------------------------;                          **MBZ CHECK**
                                        ;35168          WB_M[TEMP1].AND.R[LONLIT],       ;
U 0D9E, 0080,1002,0A3D,4047,0CDF,4      ;35169          WX.EQ.0?                        ;
                                        ;35170
                                        ;35171 =0       ;0-------------------------------;                          **MBZ CHECK**
                                        ;35172          WB_M[FPDOFFSET],                ;
U 0DF4, 0080,C592,46F0,0047,00E5,1      ;35173          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35174
                                        ;35175          ;1-------------------------------;                          **MBZ CHECK**
U 0DF5, 0586,1C93,8037,F847,00DA,7      ;35176          M[TEMP1]_MB.ANDNOT.ZLIT24[OFF]  ;
                                        ;35177
                                        ;35178          ;-------------------------------;                          **MBZ CHECK**
                                        ;35179          R[IPR.R]_M[TEMP1],              ;
U 0DA7, 0C84,1592,403C,8047,00E7,8      ;35180          NEXT/MP.MTPR.CHK.CON            ;
                                        ;35181
                                        ;35182 =00
                                        ;35183 MP.MTPR.P1BR:
                                        ;35184          ;00------------------------------;                          **MBZ CHECK**
                                        ;35185          WB_D+ZLIT16[100],               ;
U 0DF0, 0580,0D31,06F8,0047,08DF,1 377* ;35186          WB<31-30>?                      ;
                                        ;35187
                                        ;35188 MP.MTPR.PXBR:
                                        ;35189          ;01------------------------------;                          **MBZ CHECK**
                                        ;35190          WB_M[FPDOFFSET],                ;
U 0DF1, 0080,C592,46F0,0047,00E5,1      ;35191          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35192
                                        ;35193 MP.MTPR.SBR10:
                                        ;35194          ;10------------------------------;                          **MBZ CHECK**
                                        ;35195          WB_M[FPDOFFSET],                ;
U 0DF2, 0080,C592,46F0,0047,00E5,1      ;35196          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35197
                                        ;35198 =11      ;11------------------------------;                          **MBZ CHECK**
                                        ;35199          WB_M[FPDOFFSET],STEPC_2,        ; SETUP STEPC FOR CONSOL
U 0DF3, 00A0,C592,46F0,0047,017D,9      ;35200          WB<31-30>?,NEXT/MP.MTPR.DONE    ;
```

```
                                        ;35201 =0
                                        ;35202 MP.MTPR.SBR:
                                        ;35203          ;0-------------------------------;                **MBZ CHECK**
                                        ;35204          WB_D.AND.ZLIT24[OFF],           ;
U 0E30, 0580,0CB2,0A37,F847,00DF,2      ;35205          WX.EQ.0?,NEXT/MP.MTPR.SBR10     ;
                                        ;35206
                                        ;35207          ;1-------------------------------;                **MBZ CHECK**
                                        ;35208          WB_M[FPDOFFSET],                ;
U 0E31, 0080,C592,46F0,0047,00E5,1      ;35209          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35210
                                        ;35211 MP.MTPR.SLR:
                                        ;35212          ;-------------------------------;                **MBZ CHECK**
                                        ;35213          WB_M[TEMP1].ASR.P,              ;
U 0DA8, 0880,10F7,0A30,0047,08E3,4 323* ;35214          WX.EQ.0?                        ;
                                        ;35215
                                        ;35216 =0       ;0-------------------------------;                **MBZ CHECK**
                                        ;35217          WB_M[FPDOFFSET],                ;
U 0E34, 0080,C592,46F0,0047,00E5,1      ;35218          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35219
                                        ;35220          ;1-------------------------------;                **MBZ CHECK**
                                        ;35221          R[IPR.R]_M[TEMP1],              ;
U 0E35, 0C84,1592,403C,8047,00E7,8      ;35222          NEXT/MP.MTPR.CHK.CON            ;
                                        ;35223
                                        ;35224          ;********************************* CONTINUATION OF GROUP 3 REGISTERS THAT
                                        ;35225          ;********************************* TAKE MORE THAN ONE WORD TO COMPLETE.
                                        ;35226
                                        ;35227 =0
                                        ;35228 MP.MTPR.PCBB:
                                        ;35229          ;0-------------------------------;                **MBZ CHECK**
                                        ;35230          WB_M[FPDOFFSET],STEPC_2,        ; SETUP STEPC FOR CONSOL
U 0E68, 00A0,C592,46F0,0047,017D,9      ;35231          WB<31-30>?,NEXT/MP.MTPR.DONE    ;
                                        ;35232
                                        ;35233          ;1-------------------------------;                **MBZ CHECK**
                                        ;35234          WB_M[FPDOFFSET],                ;
U 0E69, 0080,C592,46F0,0047,00E5,1      ;35235          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35236
                                        ;35237 MP.MTPR.SCBB:
                                        ;35238          ;-------------------------------;                **MBZ CHECK**
                                        ;35239          WB_R[LONLIT].NOTAND.Q,           ;
U 0DAA, 0C80,003B,CA3D,4047,00E6,A      ;35240          WX.EQ.0?                        ;
                                        ;35241
                                        ;35242 =0       ;0-------------------------------;                **MBZ CHECK**
                                        ;35243          WB_M[FPDOFFSET],                ;
U 0E6A, 0080,C592,46F0,0047,00E5,1      ;35244          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35245
                                        ;35246          ;1-------------------------------;                **MBZ CHECK**
                                        ;35247          M[SCBB]_Q,                      ;
U 0E6B, 0886,E03A,403D,8047,00E7,8      ;35248          NEXT/MP.MTPR.CHK.CON            ;
```

```
                                        ;35249 MP.MTPR.ASTLVL:
                                        ;35250          ;--------------------------------;                **MBZ CHECK**
                                        ;35251          WB_M[TEMP1]+ZLIT0[3],           ; ADD 3 AND CHECK FOR CARRY INTO BIT 4.
U 0DB0, 0180,1C11,0230,1847,08DC,7 377* ;35252          WB<5-0>?                        ;
                                        ;35253
                                        ;35254 =**0111
                                        ;35255          ;**0111--------------------------;                **MBZ CHECK**
                                        ;35256          ASTLVL_M[TEMP1].RL.24,          ;
U 0DC7, 0C80,13B7,0000,0707,00E7,8      ;35257          NEXT/MP.MTPR.CHK.CON            ;
                                        ;35258
                                        ;35259          ;**1111--------------------------;
                                        ;35260          WB_M[FPDOFFSET],                ;
U 0DCF, 0080,C592,46F0,0047,00E5,1      ;35261          WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                        ;35262
                                        ;35263 MP.MTPR.SIRR:
                                        ;35264          ;--------------------------------;
U 0DB3, 0086,203A,403D,8047,00DB,A      ;35265          M[TEMP2]_Q                      ; (CON'T DO MUX/Q.S SO MOVE TO TEMP)
                                        ;35266
                                        ;35267          ;--------------------------------;
U 0DBA, 0986,2C12,0030,7847,00DB,E      ;35268          M[TEMP2]_MB.AND.ZLIT0[0F]       ; IGNORE SOURCE<31:4>
                                        ;35269
                                        ;35270          ;--------------------------------;
                                        ;35271          WB_M[TEMP2]-PL,SIZE[LONG],      ; COMPARE IPL OF THIS REQUEST WITH
U 0DBE, 0080,2B10,0B60,0047,08E0,0 410* ;35272          SIGND CMP?                      ; HIGHEST PENDING SOFTWARE REQUEST.
                                        ;35273
                                        ;35274 =00      ;00--------------------------------; NEW INTERRUPT REQUEST HAS IPL HIGHER
                                        ;35275          SOFTIPR_M[TEMP2].RR.16,         ; THAN CURRENT SOFTWARE IPL, UPDATE
U 0E00, 0880,23B7,0010,0787,00E0,2      ;35276          NEXT/MP.MTPR.SIRR10             ; HARDWARE REGISTER FOR ARBITRATION.
                                        ;35277
                                        ;35278          ;01--------------------------------; INTERRUPT ALREADY REQUESTED AT
                                        ;35279          WB_M[FPDOFFSET],STEPC_2,        ; THIS IPL, DON'T DO ANYTHING.
J 0E01, 00A0,C592,46F0,0047,017D,9      ;35280          WB<31-30>?,NEXT/MP.MTPR.DONE    ; SETUP STEPC FOR CONSOL
                                        ;35281
                                        ;35282 MP.MTPR.SIRR10:
                                        ;35283          ;10--------------------------------; LOAD PL WITH IPL OF NEW REQUEST.
U 0E02, 0880,2BC2,403D,8047,00DC,6      ;35284          PL_M[TEMP2]                     ;
                                        ;35285 =
                                        ;35286          ;--------------------------------;
                                        ;35287          M[SISR]_MB.OR.ZLITPL[1],        ; UPDATE SISR.
U 0DC6, 0186,FAD2,4030,0847,00E7,8      ;35288          NEXT/MP.MTPR.CHK.CON            ;
```

```
                                ;35289 =00
                                ;35290 =01
                                ;35291 MP.MTPR.SISR:
                                ;35292          ;01--------------------------------;
                                ;35293          M[TEMP1]_PL,                       ; PREPARE TO LOAD SOFTIPR.
U 0E05, 0486,1B37,0030,0047,00DC,E ;35294       NEXT/MP.MTPR.SISR10                ;
                                ;35295
                                ;35296 MP.MTPR.SISR05:
                                ;35297          ;10--------------------------------;
                                ;35298          PL_MSS M[SISR],                    ; GET IPL OF HIGHEST SOFTWARE INTERRUPT.
                                ;35299          WX<31-16>.NE.0?,                   ;
U 0E06, 0C80,F9C2,4DFD,8047,0CE0,5 ;35300       NEXT/MP.MTPR.SISR                  ; CHECK SISR<31:16>.NE.0
                                ;35301
                                ;35302          ;11--------------------------------;
                                ;35303          WB_M[FPDOFFSET],                   ;
U 0E07, 0080,C592,46F0,0047,00E5,1 ;35304       WB<31-30>?,NEXT/MP.MTPR.BAD.IPR    ; SOME MBZ BITS WERE NOT ZERO.
                                ;35305
                                ;35306 MP.MTPR.SISR10:
                                ;35307          ;--------------------------------;
                                ;35308          SOFTIPR_M[TEMP1].RR.16,            ; LOAD HARDWARE REGISTER FOR ARBITRATING
U 0DCE, 0080,13B7,0010,0787,00E7,8 ;35309       NEXT/MP.MTPR.CHK.CON               ; SOFTWARE INTERRUPTS.
                                ;35310
                                ;35311 =0
                                ;35312 MP.MTPR.ICCS:
                                ;35313          ;0---------------- ----------------; CATCH RETURN-1 FROM IE.ICR.SERV2
                                ;35314          M[MM.TEMP0]_MB.AND.ZLIT16[100],    ; USE CURRENT VALUES FOR TVP AND VP
U 0E6C, 0D86,DD12,0038,0047,00DE,A ;35315       NEXT/MP.MTPR.ICCS10                ;
                                ;35316
                                ;35317 MP.MTPR.ICCS05:
                                ;35318          ;1--------------------------------;
                                ;35319          WB_.NOT.(M[MM.TEMP0].RR.16),       ; BEGIN INTERVAL TIMER SERVICE.
                                ;35320          WB<5-0>?,                          ; BRANCH ON TCSR<SR,TR>.
U 0E6D, 0880,D3B6,0210,0047,0CFB,3 337* ;35321  PUSH,NEXT/IE.ICR.SERV2              ;
                                ;35322
                                ;35323 MP.MTPR.ICCS10:
                                ;35324          ;--------------------------------;
U 0DEA, 0480,002D,7030,0047,00DE,C ;35325       D_Q Q_D                            ; CAN'T DO MUX/Q.S SO MOVE TO D.
                                ;35326
                                ;35327          ;--------------------------------;
U 0DEC, 0586,2C32,0037,8847,00DE,E ;35328       M[TEMP2]_D.AND.ZLIT0[0F1]          ; CLEAR MBZ BITS IN SOURCE(EXCEPT ERROR)
                                ;35329
                                ;35330          ;--------------------------------;
U 0DEE, 0180,0C72,2030,4047,00DF,6 ;35331       D_D.AND.ZLIT28[8]                  ; LOAD D WITH ERROR BIT ONLY.
                                ;35332
                                ;35333          ;--------------------------------; LOAD USER DEFINED BITS IN
U 0DF6, 0C86,23B2,4010,0047,00E0,3 ;35334       M[TEMP2]_D.OR.(MB.RR.16)           ; COMET ORIENTATION (E,IR,IE,SC,T,R).
                                ;35335
                                ;35336          ;--------------------------------;
                                ;35337          TCSR_M[MM.TEMP0].OR.R[TEMP2],      ; TCSR_(E,TVP,IR,IE,SC,T,SR=0,TR=0,VP,R)
U 0E03, 0880,D002,4030,8167,00E7,8 ;35338       NEXT/MP.MTPR.CHK.CON               ;
```

```
                                        ;35339 =10
                                        ;35340 MP.MTPR.NICR:
                                        ;35341          ;10--------------------------------;
                                        ;35342          INIR_M[TEMP1]_Q,                   ; LOAD INTERNAL HALF OF NICR.
U 0E0E, 0886,103A,403D,8147,00E0,4      ;35343          NEXT7MP.MTPR.NICR10                ;
                                        ;35344
                                        ;35345 MP.MTPR.NICR05:
                                        ;35346          ;11--------------------------------;
                                        ;35347          WB_.NOT.(M[MM.TEMP0].RR.16),       ; BEGIN INTERVAL TIMER SERVICE.
                                        ;35348          WB<5-0>?,                          ; BRANCH ON TCSR<SR,IR>.
U 0E0F, 0880,D3B6,0210,0047,0CFB,3 337* ;35349          PUSH,NEXT/IE.ICR.SERV2             ;
                                        ;35350
                                        ;35351 MP.MTPR.NICR10:
                                        ;35352          ;--------------------------------;
U 0E04, 0086,25BE,403B,8047,00E0,D      ;35353          M[TEMP2]_R[SPNICR.SPICR]           ; SAVE SPICR.
                                        ;35354
                                        ;35355          ;--------------------------------;
U 0E0D, 0084,1592,403B,8047,00E2,9      ;35356          R[SPNICR.SPICR]_M[TEMP1]           ; LOAD SPNICR (DESTROYING SPICR).
                                        ;35357
                                        ;35358          ;--------------------------------;
                                        ;35359          R[SPNICR.SPICR].SIZ_M[TEMP2],      ; RESTORE SPICR.
U 0E29, 0482,2592,401B,8047,00E2,F      ;35360          SIZE[WORD]                         ;
                                        ;35361
                                        ;35362          ;--------------------------------;
U 0E2F, 0C80,13B6,0B50,0047,08E2,A 373* ;35363          M[TEMP1]<31-16>.EQ.FFFF?           ; TEST SPNICR FOR ALL ONES.
                                        ;35364
                                        ;35365 =10
                                        ;35366 MP.MTPR.NICR20:
                                        ;35367          ;10--------------------------------; SPNICR IS NOT ABOUT TO OVFLW CLEAR TVP
                                        ;35368          TCSR_M[MM.TEMP0].AND.ZLIT16[43],   ; RELOAD TCSR, BE SURE AND NOT
U 0E2A, 0580,DD12,0032,1967,00E7,8      ;35369          NEXT7MP.MTPR.CHK.CON               ; CLEAR TCSR<E,IR,SR,TR>.
                                        ;35370
                                        ;35371          ;11--------------------------------; SPNICR IS ABOUT TO OVERFLOW,
U 0E2B, 0186,DC92,4030,0847,00E3,6      ;35372          M[MM.TEMP0]_MB.OR.ZLIT24[1]        ; SET TVP.
                                        ;35373
                                        ;35374          ;--------------------------------;
                                        ;35375          TCSR_M[MM.TEMP0].AND.ZLIT16[143],  ; RELOAD TCSR, BE SURE AND NOT
U 0E36, 0180,DD12,003A,1967,00E7,8      ;35376          NEXT7MP.MTPR.CHK.CON               ; CLEAR TCSR<E,IR,SR,TR>.
                                        ;35377
                                        ;35378 =0
                                        ;35379 MP.MTPR.CSTS:
                                        ;35380          ;0--------------------------------; NOT SETTING BREAK
                                        ;35381          TRAR_ZLIT16[40],                   ; TRAR_1
U 0E6E, 0D80,0D37,0032,0267,00E6,7      ;35382          NEXT7MP.MTPR.TU58                  ;
                                        ;35383
                                        ;35384          ;1--------------------------------;
U 0E6F, 0180,0D37,0036,0267,00E4,4      ;35385          TRAR_ZLIT16[0C0]                   ; SEND BREAK CHARACTER
```

```
                        ;35386          ;--------------------------------;
U 0E44, 0886,2036,4030,03E7,00E4,B      ;35387          M[TEMP2]_TU58REGS               ; READ CURRENT COLD START FLAG
                        ;35388
                        ;35389          ;--------------------------------;
U 0E4B, 0980,2D12,4030,82E7,00E5,0      ;35390          TU58REGS_M[TEMP2].OR.ZLIT16[10] ; SET TU58 BREAK
                        ;35391
                        ;35392          ;--------------------------------;
                        ;35393          TU58REGS_M[TEMP2].OR.ZLIT16[40].; CLEAR TU58 BREAK
U 0E50, 018(,2D12,4032,02E7,00E6,E      ;35394          NEXT/MP.MTPR.CSTS               ;
                        ;35395
                        ;35396 MP.MTPR.TODR:
                        ;35397          ;--------------------------------;
                        ;35398          M[TEMP1]_Q,                     ; MOVE SOURCE TO MTEMP FOR SHIFTING.
U 0E52, 0C86,103A,437D,8047,00D9,F      ;35399          STEPC.GE.4? DECBY4              ; STEPC_10. (DON'T ACUTALLY BRANCH)
                        ;35400 =111
                        ;35401          ;--------------------------------;
U 0D9F, 099E,2D37,0031,0047,00E5,7      ;35402          M[TEMP2]_ZLIT16[20],DEC STEPC   ; CONSTANT FOR ENABLING MEMORY,STEPC_9.
                        ;35403
                        ;35404          ;--------------------------------;
U 0E57, 0998,0EF6,4030,0047,00E5,8      ;35405          PL_[0],DEC STEPC                ; USED TO EXTRACT DATA NIBBLE, STEPC_8.
                        ;35406
                        ;35407          ;--------------------------------;
U 0E58, 0D80,0F76,4030,A047,00E5,A      ;35408          SL_[20.]                        ; USED TO EXTRACT DATA NIBBLE.
                        ;35409
                        ;35410          ;--------------------------------;
                        ;35411          TOYCR_M[TEMP3]_ZLIT16[17],      ;
U 0E5A, 0D86,3D37,0030,B9A7,00E5,C      ;35412          NEXT/MP.MTPR.TODR.ADR           ;
                        ;35413
                        ;35414 =0
                        ;35415 MP.MTPR.TODR.LP:
                        ;35416          ;0-------------------------------;
                        ;35417          TOYCR_M[TEMP3]_MB-ZLIT16[1],    ; LOAD NEXT TOYOS ADDRESS.
U 0E70, 0586,3D10,0030,09A7,00E5,C      ;35418          NEXT/MP.MTPR.TODR.ADR           ;
                        ;35419
                        ;35420          ;1-------------------------------;
                        ;35421          STEPC_2,                        ; RESET STEPC INCASE CONSOLE MODE.
                        ;35422          WB_M[FPDOFFSET],                ; DONE WRITING TOYOS.
U 0E71, 00A0,C592,46F0,0047,017D,9      ;35423          WB<31-30>?,NEXT/MP.MTPR.DONE    ;
                        ;35424
                        ;35425 MP.MTPR.TODR.ADR:
                        ;35426          ;--------------------------------;
U 0E5C, 0480,3002,4030,81A7,00E5,E      ;35427          TOYCR_M[TEMP3].OR.R[TEMP2]      ; LATCH TOYOS ADDRESS.
                        ;35428
                        ;35429          ;--------------------------------;
U 0E5E, 0880,107E,6030,81A7,00E6,0      ;35430          TOYCR_D_(M[TEMP1].XZ).OR.R[TEMP2] ; EXTRACT DATA NIBBLE, WRITE TOYOS.
                        ;35431
                        ;35432          ;--------------------------------;
U 0E60, 0486,1277,0030,4047,00E6,2      ;35433          M[TEMP1]_(MB R[TEMP1]).RR.4     ; SETUP NEXT DATA NIBBLE
                        ;35434
                        ;35435          ;--------------------------------;
                        ;35436          TOYCR_D.XOR.ZLIT16[30],         ; DISABLE MEMORY, END TOYOS WRITE CYCLE.
U 0E62, 0980,0D33,4331,81A7,00E7,0      ;35437          DBZ STEPC?,NEXT/MP.MTPR.TODR.LP ; 8 NIBBLES WRITTEN?
```

```
                                ;35438 MP.MTPR.TU58:
                                ;35439          ;--------------------------------;
                                ;35440          TU58REGS_M[TEMP1].RR.16,        ;
U 0E67, 0080,13B7,0010,02E7,00E7,8 ;35441       NEXT/MP.MTPR.CHK.CON            ;
                                ;35442
                                ;35443 MP.MTPR.CONSOLE:
                                ;35444          ;--------------------------------;
                                ;35445          CONREGS_M[TEMP1].RR.16,         ;
U 0E8A, 0880,13B7,0010,02C7,00E7,8 ;35446       NEXT/MP.MTPR.CHK.CON            ;
                                ;35447
                                ;35448 MP.MTPR.MEMSCR:
                                ;35449          ;--------------------------------;
                                ;35450          MEMSCR_M[TEMP1].RL.24,          ; WRITE MEMORY STATUS AND CONTROL
U 0E8B, 0480,13B7,0000,0607,00E7,8 ;35451       NEXT/MP.MTPR.CHK.CON            ; REGISTER SPECIFIED BY MEMSCAR.
                                ;35452 =0
                                ;35453 MP.MTPR.TXDB:
                                ;35454          ;0--------------------------------;
                                ;35455          WB_M[TEMP2].XOR.ZLIT8[0F],      ; CHECK IF ID FIELD IS 0F.
U 0E72, 0980,2D93,4A30,7847,00E7,4 ;35456       WX.EQ.0?,NEXT/MP.MTPR.TXDB.10   ;
                                ;35457
                                ;35458          ;1--------------------------------;
                                ;35459          CRAR_ZLIT16[0],                 ; ID FIELD IS 0. TRANSMIT CHARACTER.
U 0E73, 0180,0D37,0030,0247,00E8,A ;35460       NEXT/MP.MTPR.CONSOLE            ;
                                ;35461 =0
                                ;35462 MP.MTPR.TXDB.10:
                                ;35463          ;0--------------------------------;
                                ;35464          WB_M[FPDOFFSET],                ; ID MUST BE 00 OR 0F.
U 0E74, 0080,C592,46F0,0047,00E5,1 ;35465       WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ;
                                ;35466
                                ;35467          ;1--------------------------------; ID FIELD IS 0F.
                                ;35468          WB_M[TEMP1]-ZLIT0[4],           ; DECODE DATA FIELD FOR ID FIELD = 0F.
U 0E75, 0980,1C10,0B40,2047,08E2,C 413* ;35469  SIGND CMP DEF?,SIZE[BYTE]        ;
                                ;35470 =00
                                ;35471 MP.MTPR.RESV.OPER:
                                ;35472          ;00-------------------------------; DATA > 04.
                                ;35473          WB_M[FPDOFFSET],                ;
U 0E2C, 0080,C592,46F0,0047,00E5,1 ;35474       WB<31-30>?,NEXT/MP.MTPR.BAD.IPR ; ONLY F00,F01,F02,F03, F04 ARE LEGAL.
                                ;35475
                                ;35476          ;01-------------------------------;
                                ;35477          TRAR_ZLIT16[0C0],               ; DATA = 04, CLEAR COLD START BIT.
U 0E2D, 0980,0D37,0036,0267,00E8,C ;35478       NEXT/MP.MTPR.TXDB.20            ;
                                ;35479
                                ;35480          ;10-------------------------------; DATA < 04,
                                ;35481          WB_M[TEMP1]-ZLIT0[2],           ; CHECK IF DATA = 02.
U 0E2E, 058C,1C10,0B40,1047,08E3,2 413* ;35482  SIGND CMP DEF?,SIZE[BYTE]        ;
                                ;35483 =
                                ;35484 =10      ;10-------------------------------;
U 0E32, 0C80,0036,4030,0047,00E7,8 ;35485       NEXT/MP.MTPR.CHK.CON            ; DATA = 00,01 AND 03 ARE NOPS.
                                ;35486
                                ;35487          ;11-------------------------------;
                                ;35488          RNUM_[0],                       ; ZERO TO HAVE BOOT DO UVERFY.
U 0E33, 0D81,DC37,0030,0047,0082,3 ;35489       NEXT/BO.BOOT.CLR.FLAGS          ; DATA = 02 DO A BOOT.
```

```
                                ;35490 MP.MTPR.TXDB.20:
                                ;35491          ;--------------------------------;
                                ;35492          TU58REGS_ZLIT16[0],             ;
U 0E8C, 0980,0D37,0030,02E7,00E7,8 ;35493       NEXT/MP.MTPR.CHK.CON            ;
                                ;35494 =0
                                ;35495 MP.MTPR.TBDR:
                                ;35496          ;0-------------------------------; MEMSCAR_ IS NOT NEEDED HERE ANYMORE
                                ;35497          MEMSCAR_ZLIT24[0],              ; DO NOT DISABLE BOTH HALVES, INSTEAD
U 0E76, 0980,0CB7,0030,0687,00E2,C ;35498       NEXT/MP.MTPR.RESV.OPER          ; TAKE A RESVERED OPERAND FAULT
                                ;35499
                                ;35500 MP.MTPR.TBDR.05:
                                ;35501          ;1-------------------------------;
                                ;35502          MEMSCAR_ZLIT24[3],              ;
U 0E77, 0180,0CB7,0030,1E87,00E8,B ;35503       NEXT/MP.MTPR.MEMSCR             ;
                                ;35504
                                ;35505 FREE.0E8D:
                                ;35506          ;******LOCATION NOT USED********;
U 0E8D, 0080,05B7,0030,0607,00E7,7 ;35507       MEMSCR_0,NEXT/MP.MTPR.TBDR.05   ;
                                ;35508 =00
                                ;35509 MP.MTPR.CADR:
                                ;35510          ;00------------------------------;
                                ;35511          MEMSCAR_ZLIT24[6],              ;
U 0E40, 0D80,0CB7,0030,3687,00E8,B ;35512       NEXT/MP.MTPR.MEMSCR             ;
                                ;35513
                                ;35514          ;01------------------------------;
                                ;35515          D_ZLIT4[40],                    ;
U 0E41, 0980,0DF7,2032,0047,0487,B ;35516       PUSH,NEXT/IN.VA_0               ;
                                ;35517
                                ;35518 MP.MTPR.CADR05:
                                ;35519          ;10------------------------------;
                                ;35520          VA_VA+4 CLEAR CACHE,            ;
                                ;35521          MM.ALLOW.INT?,                  ;
U 0E42, 0D81,BC11,02F0,25A7,00E4,5 ;35522       NEXT/MP.MTPR.CADR10             ;
                                ;35523
                                ;35524          ;11------------------------------;
                                ;35525          MEMSCAR_ZLIT24[6],              ;
U 0E43, 0D80,0CB7,0030,3687,00E8,B ;35526       NEXT/MP.MTPR.MEMSCR             ;
                                ;35527 =00
                                ;35528 =01
                                ;35529 MP.MTPR.CADR10:
                                ;35530          ;01------------------------------;
                                ;35531          D_D-ZLIT0[1],                   ;
U 0E45, 0980,0C30,2A30,0847,08E4,2 384* ;35532  WX.EQ.0?,NEXT/MP.MTPR.CADR05     ;
                                ;35533
                                ;35534          ;10------------------------------; CATCH RETURN-1 IF ONLY TIMER
                                ;35535          D_D-ZLIT0[1],                   ; OR CONSOLE MODE.
U 0E46, 0980,0C30,2A30,0847,08E4,2 384* ;35536  WX.EQ.0?,NEXT/MP.MTPR.CADR05     ;
                                ;35537
                                ;35538          ;11------------------------------; SERVICE INTERRUPT OR TIMER
                                ;35539          WB_M[FPDOFFSET],WB<31-30>?,     ; CHECK IF IN CONSOLE MODE.
U 0E47, 0080,C592,46F0,0047,04F6,D ;35540       PUSH,NEXT/IE.SERV.IP.TS         ;
```

```
                                ;35541 MP.MTPR.MEMERR:
                                ;35542          ;--------------------------------;
U 0E8E, 0086,2036,4030,0647,00E8,F ;35543       M[TEMP2]_MEMSCR                  ; FETCH CURRENT CONTENTS
                                ;35544
                                ;35545          ;--------------------------------;
                                ;35546          MEMSCR_M[TEMP2].ANDNOT.(R[TEMP1].RL.24), ; BIT CLEAR MEMORY REGISTER
U 0E8F, 0080,2293,8000,4607,00E7,8 ;35547       NEXT/MP.MTPR.CHK.CON             ;
                                ;35548
                                ;35549 MP.MTPR.ACCS:
                                ;35550          ;--------------------------------;
                                ;35551          FPA.ENABLE_M[TEMP1].RR.P,        ; WRITE SOURCE<15> TO FPA ENABLE.
U 0E90, 0880,1177,0030,02A7,00E7,8 ;35552       NEXT/MP.MTPR.CHK.CON             ;
                                ;35553 =0
                                ;35554 MP.MTPR.CHK.CON:
                                ;35555 MP.MTPR.IORESET-1:
                                ;35556          ;0------------------------------;
                                ;35557          WB_M[FPDOFFSET],STEPC_2,         ; SETUP STEPC FOR CONSOL
U 0E76, 00A0,C592,46F0,0047,017D,9 ;35558       WB<31-30>?,NEXT/MP.MTPR.DONE     ;
                                ;35559
                                ;35560 MP.MTPR.IORESET:
                                ;35561          ;1------------------------------; CHECK FOR INTERRUPTS
U 0E79, 0080,0036,42F0,0043,007E,8 ;35562       IO RESET,MM.ALLOW.INT?           ; (UNLESS FROM CONSOLE)
                                ;35563
                                ;35564 7E8:     ;00*******FORCE ADDRESS*********;
                                ;35565          IO RESET,                        ;
                                ;35566          M[TEMP1]_MB-ZLIT0[1],            ;
U 07E8, 0186,1C10,0A70,0843,08E7,8 378* ;35567  WX.NE.0?,NEXT/MP.MTPR.IORESET-1 ;
                                ;35568
                                ;35569 7E9:     ;01*******FORCE ADDRESS*********;
                                ;35570          M[TEMP1]_ZLIT0[400.],            ; RESTART IORESET AFTER TIMER SERVICE
U 07E9, 0086,1C37,003C,8047,00E7,9 ;35571       NEXT/MP.MTPR.IORESET             ;
                                ;35572
                                ;35573 7EA:     ;10*******FORCE ADDRESS*********;
                                ;35574          M[TEMP1]_ZLIT0[400.],            ; START OVER AGAIN IF ONLY TIMER
                                ;35575          INTPEND OR TIMER?,               ;
U 07EA, 0086,1C37,0AFC,8047,04F7,0 ;35576       PUSH,NEXT/IE.SERV.IP.TS2         ;
```

```
                                        ;35577 MP.MTPR.MME:
                                        ;35578          ;-------------------------------; SAVE IN RTEMP13 FOR THE CONSOLE.
                                        ;35579          MEMSCR_R[TEMP13]_M[TEMP1].RL.24,; WRITE MME BIT.
U 0E91, 0C84,13B7,0003,4607,00E9,4      ;35580          NEXT/MP.MTPR.FLUSH.XB           ;
                                        ;35581 =00
                                        ;35582 MP.MTPR.TBIA:
                                        ;35583          ;00-----------------------------;
                                        ;35584          TB_D+ZLIT8[4],                  ; INITIALIZE ALU, MUX, ROT, AND LITRL
U 0E48, 0581,BDB1,0030,2507,04E5,4      ;35585          PUSH,NEXT/MP.INVALIDATE.TB      ; THESE FIELDS ARE THE SAME IN THE LOOP
                                        ;35586
                                        ;35587          ;01-----------------------------;
                                        ;35588          VA_D_ZLIT24[80],CLEAR FLAG1,    ; INVALIDATE SYSTEM SPACE TB.
                                        ;35589          MM.ALLOW.INT?,                  ; CHECK IPEND OR TS (IN CSA<1>).
U 0E49, 0508,0CB7,22F4,04A7,04E4,C      ;35590          PUSH,NEXT/MP.MTPR.TBIA20        ; PUSH FOR JUMP TO MP.INVALIDATE.TB
                                        ;35591
                                        ;35592          ;10-----------------------------; CATCH RETURN FROM MP.INVALIDATE.TB
U 0E4A, 0481,A002,403D,8487,00E7,8      ;35593          PC_M[PC],NEXT/MP.MTPR.CHK.CON   ; FLUSH THE XB.
                                        ;35594 =
                                        ;35595 =00
                                        ;35596 MP.MTPR.TBIA20:
                                        ;35597          ;00-----------------------------;
                                        ;35598          TB_D+ZLIT8[4],                  ; INITIALIZE ALU, MUX, ROT, AND LITRL
U 0E4C, 0D81,BDB1,0030,2507,00E5,4      ;35599          NEXT/MP.INVALIDATE.TB           ; THESE FIELDS ARE THE SAME IN THE LOOP
                                        ;35600
                                        ;35601 =10      ;10-----------------------------;
                                        ;35602          INTPEND OR TIMER?,              ; SERVICE THE INTERRUPT OR TIMER.
U 0E4E, 0C80,0036,4AF0,0047,00F7,0      ;35603          NEXT/IE.SERV.IP.TS2             ;
                                        ;35604 =
                                        ;35605 FREE.0E92:
                                        ;35606          ;******LOCATION NOT USED********;
                                        ;35607          WB_Q,                           ; SET CONDITION CODES.
U 0E92, 0C80,003A,403D,8047,00E7,8      ;35608          NEXT/MP.MTPR.CHK.CON            ;
                                        ;35609
                                        ;35610 MP.MTPR.TBIS:
                                        ;35611          ;-------------------------------;
U 0E93, 0C81,B58E,403D,8507,00E9,4      ;35612          TB_R[ZERO]                      ; INVALIDATE TB ENTRY
                                        ;35613
                                        ;35614 MP.MTPR.FLUSH.XB:
                                        ;35615          ;-------------------------------;
                                        ;35616          FLUSH_XB,                       ;
U 0E94, 0481,A592,4030,0487,00E7,8      ;35617          NEXT/MP.MTPR.CHK.CON            ;
                                        ;35618
                                        ;35619 MP.MTPR.TBDATA:
                                        ;35620          ;-------------------------------;
U 0E95, 0884,1477,0030,4047,00E9,6      ;35621          R[TEMP1]_M[TEMP1].RL.9          ; ROTATE FOR LOADING INTO TB.
                                        ;35622
                                        ;35623          ;-------------------------------;
                                        ;35624          TB_R[TEMP1],                    ; LOAD DATA INTO THE TB.
U 0E96, 0081,B58E,4030,4507,00E7,8      ;35625          NEXT/MP.MTPR.CHK.CON            ;
```

```
                                ;35626 17FB:
                                ;35627 MP.MTPR.TBCK:
                                ;35628          ;11*******FORCE ADDRESS*********;
U 17FB, 0480,003A,402D,94A7,017F,C ;35629       VA_Q,SIZE[LONG],CCOP2           ; LOAD VA FOR DOING PROBE AND SET CC.
                                ;35630
                                ;35631 17FC:    ;*********FORCE ADDRESS*********;
U 17FC, 0C80,0036,4780,005F,097D,E 479* ;35632  PROBE READ?,SIZE[BYTE]           ; CHECK FOR A TB HIT.
                                ;35633
                                ;35634 17DE:    ;1110*****FORCE ADDRESS*********;
                                ;35635          WB_M[FPDOFFSET],                ; NOT A HIT LEAVE V BIT CLEAR
U 17DE, 0880,C592,46F0,0047,00E4,D ;35636       WB<31-30>?,NEXT/MP.MTPR.EXIT    ;
                                ;35637
                                ;35638 17DF:    ;1111*****FORCE ADDRESS*********;
                                ;35639          SET V,WB_M[FPDOFFSET],          ; TB HIT, SET THE V BIT
U 17DF, 0C80,C592,46F0,08E7,00E4,D ;35640       WB<31-30>?,NEXT/MP.MTPR.EXIT    ;
                                ;35641 =01
                                ;35642 MP.MTPR.EXIT:                            ; LABEL USED BY IANDE
                                ;35643          ;01-----------------------------;
                                ;35644          M[FPDOFFSET]_R[ZERO],           ; CLEAR FLAGS IN FPDOFFSET FOR IANDE.
U 0E4D, 0086,C5BE,413D,8047,003F,9 ;35645       IRD1                            ;
                                ;35646
                                ;35647 MP.MTPR.RESET.STEPC:
                                ;35648          ;11-----------------------------;
                                ;35649          DEC STEPC,                      ; RESET STEPC FOR CONSOLE TYPING.
                                ;35650          R[TEMP12]_PSL,                  ; SAVE PSL FOR CONSOLE.
U 0E4F, 089C,0036,4033,0087,008A,F ;35651       NEXT/CN.PROMPT                  ; RETURN TO CONSOLE MODE FLOWS.
                                ;35652 17D9:
                                ;35653 MP.MTPR.DONE:
                                ;35654          ;01*******FORCE ADDRESS*********;
U 17D9, 0080,003A,412D,9047,003F,9 ;35655       WB_Q,CCOP2,SIZE[LONG],IRD1      ; SET CONDITION CODES ON SOURCE
                                ;35656
                                ;35657 17DB:    ;11*******FORCE ADDRESS*********;
                                ;35658          DEC STEPC,                      ; RESET STEPC FOR CONSOLE TYPING.
                                ;35659          R[TEMP12]_PSL,                  ; SAVE PSL FOR CONSOLE.
U 17DB, 089C,0036,4033,0087,008A,F ;35660       NEXT/CN.PROMPT                  ; RETURN TO CONSOLE MODE FLOWS.
                                ;35661
                                ;35662 =01
                                ;35663 MP.MTPR.BAD.IPR:
                                ;35664          ;01-----------------------------;
                                ;35665          CLEAR FLAG3,                    ; PUSH PCBACK-2
U 0E51, 0418,0036,4030,0047,00FF,8 ;35666       NEXT/IE.OPER.FAULT              ; TAKE A RESERVED OPERAND FAULT.
                                ;35667
                                ;35668          ;11-----------------------------;
                                ;35669          M[TEMP7]_ZLIT24[11],STEPC_2,    ;
U 0853, 05A6,7CB7,0030,8847,0096,C ;35670       NEXT/CN.ERROR                   ; RETURN TO CONSOLE MODE FLOWS.
```

```
                                ;35671 .TOC "  MTPR, MFPR, CHMX                 : MP.INVALIDATE.TB"
                                ;35672
                                ;35673 ;*****************************************************************************
                                ;35674 ;       Entry points    MP.INVALIDATE.TB
                                ;35675 ;
                                ;35676 ;       Input           VA & D          Address to begin invalidating at.
                                ;35677 ;                       STEPC           Count of double TB entries to invalidate
                                ;35678 ;                       FLAG1   If clear will invalidate 32+STEPC entries
                                ;35679 ;
                                ;35680 ;       Output          FLAG1           Set always
                                ;35681 ;                       STEPC           Zero always
                                ;35682 ;
                                ;35683 ;       Resources       VA & D          Used to address TB
                                ;35684 ;
                                ;35685 ;       NOTE: It is necessary for proper invalidation that D<8> = 0.
                                ;35686 ;       Also the cycle that calls this routine MUST have the same
                                ;35687 ;       ALU, MUX, ROT, LIT, and LITRL fields as the invalidation
                                ;35688 ;       loop instruction, it cannot however be used as part
                                ;35689 ;       of the invalidation.
                                ;35690 ;
                                ;35691 ;*****************************************************************************
                                ;35692
                                ;35693 =00
                                ;35694 MP.INVALIDATE.TB:
                                ;35695         ;00-------------------------------;
                                ;35696         VA_D_D+ZLIT8[4] INVALIDATE TB,  ; INCREMENT VA IN BIT 10.
                                ;35697         DBZ STEPC?,                     ;
U 0E54, 0180,0DB1,2330,2536,00E5,4  ;35698         NEXT/MP.INVALIDATE.TB           ; LOOP.
                                ;35699
                                ;35700         ;01-------------------------------;
                                ;35701         TB_D+ZLIT8[4],                  ;
                                ;35702         SET FLAG1,                      ;
U 0E55, 0549,BDB1,05F0,2507,00E5,4  ;35703         FLAG1?,NEXT/MP.INVALIDATE.TB    ; DO LOOP TWICE.
                                ;35704
                                ;35705         ;10-------------------------------;
U 0E56, 0830,0036,40B0,0047,0000,1  ;35706         RETURN [+1]                     ;
                                ;35707 =
```

```
                        ;35708 .TOC "  MTPR, MFPR, CHMX                  : MFPR"
                        ;35709
                        ;35710 ;*****************************************************************************
                        ;35711 ;       MFPR    procreg.rl, dst.wl
                        ;35712 ;
                        ;35713 ;       Input           MDR             Source processor register number
                        ;35714 ;
                        ;35715 ;       Resources       MM.TEMP0        Temp when reading interval timer regs
                        ;35716 ;                       TEMP1           Pass register # to SELECT.IPR
                        ;35717 ;                       TEMP2           Temporary
                        ;35718 ;                       MSCAR
                        ;35719 ;                       PL
                        ;35720 ;                       SL
                        ;35721 ;                       LONLIT
                        ;35722 ;
                        ;35723 ;       Subroutines     SELECT.IPR
                        ;35724 ;                       IE.OPER.FAULT
                        ;35725 ;                       IE.ICR.SERV2
                        ;35726 ;                       OS.WRT1
                        ;35727 ;                       IL.MOV.B.W.L.MEM        Share writing of destination
                        ;35728 ;
                        ;35729 ;       MFPR IS BROKEN INTO 3 SECTIONS.
                        ;35730 ;       FOR GROUP 1 THE REGISTER NUMBER CORRESPONDS WITH RNUM FOR IPR(RNUM).
                        ;35731 ;       FOR GROUP 2 SP (THE CURRENT STACK POINTER) SHOULD BE USED
                        ;35732 ;               INSTEAD OF KSP OR ISP.
                        ;35733 ;       FOR GROUP 3 A WIDE BRANCH MUST BE DONE ON THE REGISTER NUMBER.
                        ;35734 ;
                        ;35735 ;       CHARLIE'S FLOWS 3.2
                        ;35736 ;*****************************************************************************
                        ;35737
                        ;35738 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;35739 MP.MFPR.FPA:
                        ;35740          ;-------------------------------;
U 0505, 0780,0C37,0030,0907,0039,8      ;35741          STEPC_ZLITO[1]                  ; LOAD STEPC TO INDICATE PRESENCE OF FPA
                        ;35742
                        ;35743 =000
                        ;35744 MP.MFPR:
                        ;35745          ;000-----------------------------;
U 0598, 0885,2592,4780,4207,0CDA,C 479* ;35746          R[TEMP1]_M[MDR],PSL<IS.CURM>?,  ; DECODE THE REGISTER NUMBER INTO
                        ;35747          PUSH,NEXT/MP.SELECT.IPR         ; ONE OF THREE GROUPS.
                        ;35748
                        ;35749          ;001-----------------------------;
                        ;35750          MDR_0,                          ; USED TO CLEAR Q FOR CONSOLE
U 0399, 0480,0036,4030,04E7,00DA,6      ;35751          NEXT/MP.MFPR.CHK.CON            ;
                        ;35752
                        ;35753 MP.MFPR.WRITE:
                        ;35754          ;010-----------------------------;
                        ;35755          MDR_0,                          ; USED TO CLEAR Q FOR CONSOLE
U 039A, 0480,0036,4030,04E7,00DA,6      ;35756          NEXT/MP.MFPR.CHK.CON            ;
```

```
                                ;35757          ;011----------------------------;
                                ;35758          MDR_0,                          ; USED TO CLEAR Q FOR CONSOLE
                                ;35759          WB_M[TEMP1],WB<5-0>?,           ; GROUP 3, DO A WIDE BRANCH.
U 039B, 0080,1592,4230,04E7,00E0,0      ;35760  NEXT/MP.MFPR.BRANCH             ;
                                ;35761
                                ;35762          ;100----------------------------; BAD IPR NUMBER FROM MP.SELECT.IPR
                                ;35763          WB_M[FPDOFFSET],                ;
U 039C, 0080,C592,46F0,0047,00E5,D      ;35764  WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                ;35765 =
                                ;35766 .REGION/MPRCHM.R1L,MPRCHM.R1H/MPRCHM.R2L,MPRCHM.R2H/MPRCHM.R3L,MPRCHM.R3H
                                ;35767 =000000
                                ;35768 MP.MFPR.BRANCH:
                                ;35769 =010000 ;010000----------------------------; REGISTER NUMBER 10 = PCBB.
                                ;35770          M[TEMP2]_R[PCBB],               ;
U 0E10, 0C86,25BE,4039,4047,00DA,6      ;35771  NEXT/MP.MFPR.CHK.CON            ;
                                ;35772
                                ;35773          ;010001----------------------------; REGISTER NUMBER 11 = SCBB.
                                ;35774          R[TEMP2]_M[SCBB],               ;
U 0E11, 0C84,E592,4030,8047,00DA,6      ;35775  NEXT/MP.MFPR.CHK.CON            ;
                                ;35776
                                ;35777          ;010010----------------------------; REGISTER NUMBER 12 = IPL.
                                ;35778          M[TEMP2]_PSL.PL_[16.],          ;
U 0E12, 0986,2EF6,4030,8087,00E9,7      ;35779  NEXT/MP.MFPR.IPL                ;
                                ;35780
                                ;35781          ;010011----------------------------; REGISTER NUMBER 13 = ASTLVL.
                                ;35782          M[TEMP2]_ASTLVL.PL_[24.],       ;
U 0E13, 0086,2EF6,4030,C747,00E9,9      ;35783  NEXT/MP.MFPR.ASTLVL             ;
                                ;35784
                                ;35785          ;010100----------------------------; REGISTER NUMBER  14 = SIRR(WRITE ONLY)
                                ;35786          WB_M[FPDOFFSET],                ;
U 0E14, 0080,C592,46F0,0047,00E5,D      ;35787  WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                ;35788
                                ;35789          ;010101----------------------------; REGISTER NUMBER 15 = SISR.
                                ;35790          R[TEMP2]_M[SISR],               ;
U 0E15, 0884,C592,4030,8047,00DA,6      ;35791  NEXT/MP.MFPR.CHK.CON            ;
                                ;35792
                                ;35793          ;010110----------------------------; REGISTER NUMBER 16 IS RESERVED.
                                ;35794          WB_M[FPDOFFSET],                ;
U 0E16, 0080,C592,46F0,0047,00E5,D      ;35795  WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
```

```
                                        ;35796          ;010111------------------------; REGISTER NUMBER 17 = CMIERR.
                                        ;35797          M[TEMP2] ZLIT0[0],              ; CLEAR TEMP TO BUILD RESULT IN.
U 0E17, 0186,2C37,0030,0047,00DA,0      ;35798          NEXT/MP.MFPR.CMIERR             ;
                                        ;35799
                                        ;35800          ;011000------------------------; REGISTER NUMBER 18 = ICCS.
                                        ;35801          M[TEMP2] TCSR.IICR,PL_[16.],    ;
U 0E18, 0186,2EF6,4030,81E7,00E9,E      ;35802          NEXT/MP.MFPR.ICCS               ;
                                        ;35803
                                        ;35804          ;011001------------------------; REGISTER NUMBER 19 = NICR(WRITE ONLY)
                                        ;35805          WB_M[FPDOFFSET],                ;
U 0E19, 0080,C592,46F0,0047,00E5,D      ;35806          WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                        ;35807
                                        ;35808          ;011010------------------------; REGISTER NUMBER 1A = ICR.
                                        ;35809          M[MM.TEMP0] TCSR.IICR,          ;
J 0E1A, 0086,D036,4030,01E7,00E7,B      ;35810          NEXT/MP.MFPR.ICR                ;
                                        ;35811
                                        ;35812          ;011011------------------------; REGISTER NUMBER 1B = TODR.
                                        ;35813          M[TEMP0] ZLIT0[6],STEPC_14.,    ; RETRY COUNT IS 6, STEPC IS FOR LOOPING
J 0E1B, 09B6,0C37,0030,3047,00E7,D      ;35814          NEXT/MP.MFPR.TODR               ;
                                        ;35815
                                        ;35816          ;011100------------------------; REGISTER NUMBER 1C =  CSRS.
                                        ;35817          TRAR_ZLIT16[80],                ; CRAR_2, PL_0
U 0E1C, 0D80,0037,0034,0267,00EB,0      ;35818          NEXT7MP.MFPR.TU58               ;
                                        ;35819
                                        ;35820          ;011101------------------------; REGISTER NUMBER 1D = CSRD.
                                        ;35821          TRAR_ZLIT16[0],                 ;
U 0E1D, 0980,0D37,0030,0267,00EB,0      ;35822          NEXT7MP.MFPR.TU58               ;
                                        ;35823
                                        ;35824          ;011110------------------------; REGISTER NUMBER 1E = CSTS.
                                        ;35825          TRAR_ZLIT16[40],                ; TRAR_1, PL_0
U 0E1E, 0D80,0D37,0032,0267,00EB,0      ;35826          NEXT7MP.MFPR.TU58               ;
                                        ;35827
                                        ;35828          ;011111------------------------; REGISTER NUMBER 1F = CSTD (WRITE ONLY)
                                        ;35829          WB_M[FPDOFFSET],                ;
U 0E1F, 0080,C592,46F0,0047,00E5,D      ;35830          WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                        ;35831
                                        ;35832          ;100000------------------------; REGISTER NUMBER 20 = RXCS.
                                        ;35833          CRAR_ZLIT16[80],                ; CRAR_2, PL_0
U 0E20, 0D80,0D37,0034,0247,00EB,1      ;35834          NEXT7MP.MFPR.CONSOLE            ;
                                        ;35835
                                        ;35836          ;100001------------------------; REGISTER NUMBER 21 = RXDB.
                                        ;35837          CRAR_ZLIT16[0],                 ;
U 0E21, 0980,0D37,0030,0247,00EB,1      ;35838          NEXT7MP.MFPR.CONSOLE            ;
                                        ;35839
                                        ;35840          ;100010------------------------; REGISTER NUMBER 22 = TXCS.
                                        ;35841          CRAR_ZLIT16[40],                ; CRAR_1, PL_0
U 0E22, 0D80,0D37,0032,0247,00EB,1      ;35842          NEXT7MP.MFPR.CONSOLE            ;
                                        ;35843
                                        ;35844          ;100011------------------------; REGISTER NUMBER 23 = TXDB(WRITE ONLY)
                                        ;35845          WB_M[FPDOFFSET],                ;
U 0E23, 0080,C592,46F0,0047,00E5,D      ;35846          WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                        ;35847
                                        ;35848          ;100100------------------------; REGISTER NUMBER 24 = TBDR.
                                        ;35849          MEMSCAR_ZLIT24[3],              ;
U 0E24, 0980,0CB7,0030,1E87,00EB,3      ;35850          NEXT/MP.MFPR.MEMSCR             ;
```

```
                                        ;35851          ;100101------------------------; REGISTER NUMBER 25 = CADR.
                                        ;35852          MEMSCAR_ZLIT24[6],              ;
U 0E25, 0580,0CB7,0030,3687,00EB,3      ;35853          NEXT/MP.MFPR.MEMSCR             ;
                                        ;35854
                                        ;35855          ;100110------------------------; REGISTER NUMBER 26 = MCESR.
                                        ;35856          MEMSCAR_ZLIT24[8],              ;
U 0E26, 0D80,0CB7,0030,4687,00EB,3      ;35857          NEXT/MP.MFPR.MEMSCR             ;
                                        ;35858
                                        ;35859          ;100111------------------------; REGISTER NUMBER 27 = CAER.
                                        ;35860          MEMSCAR_ZLIT24[4],              ;
U 0E27, 0180,0CB7,0030,?687,00EB,3      ;35861          NEXT/MP.MFPR.MEMSCR             ;
                                        ;35862
                                        ;35863          ;101000------------------------; REGISTER NUMBER 28 = ACCS.
                                        ;35864          FPA PRESENT?,                   ;
U 0E28, 0C80,0036,4C30,0047,00FF,E      ;35865          NEXT/MP.MFPR.ACCS               ;
                                        ;35866
                                        ;35867 =110111  ;110111------------------------; REGISTER NUMBERS 29 TO 37 ARE RESERVED
                                        ;35868          WB_M[FPDOFFSET],                ;
U 0E37, 0080,C592,46F0,0047,00E5,D      ;35869          WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                        ;35870
                                        ;35871          ;111000------------------------;
                                        ;35872          MEMSCAR_ZLIT24[0],              ; REGISTER NUMBER 38 = MME.
U 0E38, 0180,0CB7,0030,0687,00EB,4      ;35873          NEXT/MP.MFPR.MME                ;
                                        ;35874
                                        ;35875          ;111001------------------------; REGISTER NUMBER 39 = TBIA(WRITE ONLY)
                                        ;35876          WB_M[FPDOFFSET],                ;
U 0E39, 0080,C592,46F0,0047,00E5,D      ;35877          WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                        ;35878
                                        ;35879          ;111010------------------------; REGISTER NUMBER 3A = TBIS(WRITE ONLY)
                                        ;35880          WB_M[FPDOFFSET],                ;
U 0E3A, 0080,C592,46F0,0047,00E5,D      ;35881          WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                        ;35882
                                        ;35883          ;111011------------------------; REGISTER NUMBER 3B = TBDATA.
                                        ;35884          VA_R[P0BR],                     ; LOAD ADDRESS FOR READING TB.
U 0E3B, 0480,05BE,403A,04A7,00EB,5      ;35885          NEXT/MP.MFPR.TBDATA             ;
                                        ;35886
                                        ;35887          ;111100------------------------; REGISTER NUMBER 3C IS RESERVED.
                                        ;35888          WB_M[FPDOFFSET],                ;
U 0E3C, 0080,C592,46F0,0047,00E5,D      ;35889          WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                        ;35890
                                        ;35891          ;111101------------------------; REGISTER NUMBER 3D = PMR.
                                        ;35892          M[TEMP1]_R[PCBB],               ; GET BASE ADDRESS OF
U 0E3D, 0C86,15BE,4039,4047,00EB,8      ;35893          NEXT/MP.MFPR.PMR                ; PROCESS CONTROL BLOCK.
                                        ;35894
                                        ;35895          ;111110------------------------; REGISTER NUMBER 3E = SID (READ ONLY).
                                        ;35896          D_ZLIT24[2] M[TEMP1]_HARD.REV,  ; LOAD PROCESSOR ID, AND HARDWARE REV
U 0E3E, 0D86,1CB6,6030,1227,00EB,8      ;35897          NEXT/MP.MFPR.SID                ;
                                        ;35898
                                        ;35899          ;111111------------------------; REGISTER NUMBER 3F IS RESERVED.
                                        ;35900          WB_M[FPDOFFSET],                ;
U 0E3F, 008?,C592,46F0,0047,00E5,D      ;35901          WB<31-30>?,NEXT/MP.MFPR.BAD.IPR ;
                                        ;35902
                                        ;35903 MP.MFPR.IPL:
                                        ;35904          ;--------------------------------;
U 0E97, 0580,0F76,4030,2847,00E9,8      ;35905          SL_[5]                          ;
```

```
                                ;35906 MP.MFPR.XTR.WRITE:
                                ;35907          ;--------------------------------;
                                ;35908          R[TEMP2]_M[TEMP2].XZ,           ;
U 0E98, 0C84,2077,0030,8047,00DA,6 ;35909       NEXT/MP.MFPR.CHK.CON            ;
                                ;35910
                                ;35911 MP.MFPR.ASTLVL:
                                ;35912          ;--------------------------------;
                                ;35913          SL_[3],
U 0E99, 0980,0F76,4030,1847,00E9,8 ;35914       NEXT/MP.MFPR.XTR.WRITE          ;
                                ;35915 =000
                                ;35916 MP.MFPR.CMIERR:
                                ;35917          ;000-----------------------------;
                                ;35918          MEMSCAR_ZLIT24[0E],             ; LOAD ADDRESS OF CMI DISABLE REGISTER.
U 0DA0, 0180,0CB7,0030,7687,04E9,A ;35919       PUSH,NEXT/MP.MFPR.BUILD.CMIERR  ;
                                ;35920
                                ;35921          ;001-----------------------------;
                                ;35922          MEMSCAR_ZLIT24[1],              ; LOAD ADDRESS OF SAVED MODE REGISTER.
U 0DA1, 0D80,0CB7,0030,0E87,04E9,A ;35923       PUSH,NEXT/MP.MFPR.BUILD.CMIERR  ;
                                ;35924
                                ;35925          ;010-----------------------------;
                                ;35926          MEMSCAR_ZLIT24[2],              ; LOAD ADDRESS OF READ LOCK TIMEOUT BIT.
U 0DA2, 0D80,0CB7,0030,1687,04E9,A ;35927       PUSH,NEXT/MP.MFPR.BUILD.CMIERR  ;
                                ;35928
                                ;35929          ;011-----------------------------;
                                ;35930          MEMSCAR_ZLIT24[0D],             ; LOAD ADDRESS OF TB GROUP PARITY ERR.
U 0DA3, 0180,0CB7,0030,6E87,04E9,A ;35931       PUSH,NEXT/MP.MFPR.BUILD.CMIERR  ;
                                ;35932
                                ;35933          ;100-----------------------------;
                                ;35934          MEMSCAR_ZLIT24[0C],             ; LOAD ADDRESS OF TB HIT REGISTER.
U 0DA4, 0580,0CB7,0030,6687,04E9,A ;35935       PUSH,NEXT/MP.MFPR.BUILD.CMIERR  ;
                                ;35936
                                ;35937          ;101-----------------------------;
                                ;35938          MEMSCAR_ZLIT24[9],              ; LOAD ADDRESS OF BUS ERROR REGISTER.
U 0DA5, 0980,0CB7,0030,4E87,04E9,A ;35939       PUSH,NEXT/MP.MFPR.BUILD.CMIERR  ;
                                ;35940
                                ;35941 MP.MFPR.CHK.CON:
                                ;35942          ;110-----------------------------;
                                ;35943          WB_M[FPDOFFSET],                ;
U 0DA6, 0880,C592,46F0,0047,00E5,9 ;35944       WB<31-30>?,NEXT/MP.MFPR.DONE    ; CHECK IF IN CONSOLE MODE.
                                ;35945 =
                                ;35946 MP.MFPR.BUILD.CMIERR:
                                ;35947          ;--------------------------------;
U 0E9A, 0086,1036,4030,0647,00E9,B ;35948       M[TEMP1]_MEMSCR                 ; LOAD DESIRED STATUS REG INTO MTEMP.
                                ;35949
                                ;35950          ;--------------------------------;
U 0E9B, 0886,13B7,0020,0047,00E9,C ;35951       M[TEMP1]_MB.RL.8                ; ROTATE DATA FROM <27:24> TO <3:0>.
                                ;35952
                                ;35953          ;--------------------------------; ROTATE DESTINATION TEMP TO MAKE ROOM
U 0E9C, 0086,2237,0030,8047,00E9,D ;35954       M[TEMP2]_MB.RL.4                ; FOR NEXT 4 BIT STATUS REGISTER.
                                ;35955
                                ;35956          ;--------------------------------; EXTRACT NEXT 4 BIT REGISTER AND
                                ;35957          R[TEMP2]_(RB.OR.M[TEMP1]<3-0>), ; OR INTO LOW NIBBLE OF DESTINATION.
U 0E9D, 0884,13FE,40B0,8047,0000,1 ;35958       RETURN [+1]                     ;
```

```
                                ;35959 MP.MFPR.ICCS:
                                ;35960          ;--------------------------------;
U 0E9E, 0886,20F7,0030,0047,00E9,F ;35961       M[TEMP2]_MB.ASR.P               ; SHIFT ICCS INTO LOWER WORD LEAVING
                                ;35962                                          ; THE ERROR BIT IN BIT 31.
                                ;35963          ;--------------------------------;
U 0E9F, 0780,03FF,FFF9,F047,00EA,0 ;35964       LONLIT_[800000C1]               ; LOAD MASK FOR MBZ BITS.
                                ;35965
                                ;35966          ;--------------------------------;
                                ;35967          M[TEMP2]_MB.AND.R[LONLIT],      ; CLEAR MBZ BITS IN ICCS.
U 0EA0, 0886,2002,003D,4047,00DA,6 ;35968       NEXT/MP.MFPR.CHK.CON            ;
                                ;35969
                                ;35970 =0       ;0--------------------------------;
                                ;35971          M[TEMP2]_R[SPNICR.SPICR].RR.16, ; CATCH RETURN -1 FROM INTERVAL TIMER
U 0E7A, 0486,22B7,001B,8047,00EA,1 ;35972       NEXT/MP.MFPR.ICR10              ;
                                ;35973
                                ;35974 MP.MFPR.ICR:
                                ;35975          ;1--------------------------------;
                                ;35976          WB_.NOT.(M[MM.TEMP0].RR.16),    ; BEGIN INTERVAL TIMER SERVICE.
                                ;35977          WB<5-0>?,                       ; BRANCH ON TCSR<SR,TR>.
U 0E7B, 0880,D3B6,0210,0047,0CFB,3 337* ;35978  PUSH,NEXT/IE.ICR.SERV2           ;
                                ;35979
                                ;35980 MP.MFPR.ICR10:
                                ;35981          ;--------------------------------;
                                ;35982          R[TEMP2].SIZ_M[MM.TEMP0],       ;
U 0EA1, 0082,D592,4010,8047,00EA,2 ;35983       SIZE[WORD]                      ;
                                ;35984
                                ;35985          ;--------------------------------;
                                ;35986          M[TEMP2]_MB-ZLIT16[1],          ; CORRECT FOR SPICR=ICR<31:16>+1
U 0EA2, 0186,2D10,0030,0847,00DA,6 ;35987       NEXT/MP.MFPR.CHK.CON            ;
                                ;35988 =0
                                ;35989 MP.MFPR.TODR.RETRY:
                                ;35990          ;0--------------------------------;
                                ;35991          M[TEMP2]_R[ZERO],               ; TO MANY RETRIES.
U 0E7C, 0886,25BE,403D,8047,00DA,6 ;35992       NEXT/MP.MFPR.CHK.CON            ;
                                ;35993
                                ;35994 MP.MFPR.TODR:
                                ;35995          ;1--------------------------------;
U 0E7D, 0886,45BE,403D,8047,00EA,3 ;35996       M[TEMP4]_R[ZERO]                ; ZERO TEMP4 TO BUILD TOYOS
                                ;35997
                                ;35998          ;--------------------------------;
                                ;35999          TOYCR_M[TEMP3]_ZLIT16[17],      ; SETUP ADDRESS
U 0EA3, 059E,3D37,0030,B9A7,00EA,4 ;36000       DEC STEPC                       ; STEPC_13 FOR FIRST PASS
                                ;36001
                                ;36002 MP.MFPR.TODR.LP:
                                ;36003          ;--------------------------------;
U 0EA4, 0580,3D12,4031,01A7,00EA,5 ;36004       TOYCR_M[TEMP3].OR.ZLIT16[20]    ; LATCH TOYOS ADDRESS
                                ;36005
                                ;36006          ;--------------------------------;
U 0EA5, 0980,0F76,4030,2047,00EA,6 ;36007       SL_[4]                          ; WAIT FOR DATA TO STABALIZE.
                                ;36008
                                ;36009          ;--------------------------------;
U 0EA6, 0886,1036,4030,0127,00EA,7 ;36010       M[TEMP1]_TOYOS.TOYCN            ; READ DATA
```

```
                        ;36011          ;--------------------------------;
U 0EA7, 0084,107E,4031,0047,00EA,8      ;36012          R[TEMP4]_RB.OR.(M[TEMP1].XZ)    ;
                        ;36013
                        ;36014          ;--------------------------------;
U 0EA8, 0886,13B7,0010,0047,00EA,9      ;36015          M[TEMP1]_MB.RR.16               ; MOVE TOYCN DATA TO BITS <15:0>
                        ;36016
                        ;36017          ;--------------------------------;
                        ;36018          R[TEMP5].SIZ_M[TEMP1],          ; BUILD TOYCN IN RTEMP5 1 BYTE AT A TIME
                        ;36019          SIZE[BYTE],                     ;
                        ;36020          STEPC.GE.4? DECBY4,             ; STEPC = 13,9,5,1,  14,10,6,2
U 0EA9, 0482,1592,4341,4047,00DB,0      ;36021          NEXT/MP.MFPR.TODR.99            ;
                        ;36022 =000
                        ;36023 MP.MFPR.TODR.99:
                        ;36024 =001
                        ;36025          ;001-----------------------------;
                        ;36026          M[TEMP6]_R[TEMP5],              ; SAVE PASS 1 TOYCN
U 0DB1, 0CB6,65BE,4031,4047,00EA,A      ;36027          STEPC_14.,NEXT/MP.MFPR.TODR.20  ; SETUP STEPC FOR PASS 2
                        ;36028
                        ;36029          ;010-----------------------------;
                        ;36030          WB_M[TEMP5].XOR.R[TEMP6],       ; COMPARE PASS1 AND PASS2 TOYCN VALUES.
U 0DB2, 0C80,5003,4A31,8047,00E7,E      ;36031          WX.EQ.0?,NEXT/MP.MFPR.TODR.CHK  ;
                        ;36032
                        ;36033 =100     ;100-----------------------------;
                        ;36034          M[TEMP5]_MB.RL.8,               ; PREPARE FOR NEXT TOYCN BYTE
U 0DB4, 0C86,53B7,0020,0047,00EA,A      ;36035          NEXT/MP.MFPR.TODR.20            ;
                        ;36036
                        ;36037          ;101-----------------------------;
                        ;36038          WB_M[FPDOFFSET],WB<31-30>?,     ; CHECK IF IN CONSOLE MODE.
U 0DB5, 0080,C592,46F0,0047,04F6,D      ;36039          PUSH,NEXT/IE.SERV.IP.TS         ;
                        ;36040 =
                        ;36041 MP.MFPR.TODR.20:
                        ;36042          ;--------------------------------;
U 0EAA, 0486,4277,0031,0047,00EA,B      ;36043          M[TEMP4]_MB.RP.4                ; PREPARE FOR NEXT TOYOS NIBBLE
                        ;36044
                        ;36045          ;--------------------------------;
                        ;36046          TOYCR_M[TEMP3]_MB-ZLIT16[1],    ; SETUP NEXT TOYOS ADDRESS
U 0EAB, 0D86,3D10,0030,09A7,00EA,4      ;36047          NEXT/MP.MFPR.TODR.LP            ;
                        ;36048 =0
                        ;36049 MP.MFPR.TODR.CHK:
                        ;36050          ;0-----------------------------;
                        ;36051          M[TEMP0]_MB-ZLIT0[1],STEPC_14., ; DECREMENT RETRY COUNT, SETUP STEPC FOR
                        ;36052          WX.NE.0?,                       ;
U 0E7E, 05B6,0C10,0A70,0847,08E7,C 378* ;36053          NEXT/MP.MFPR.TODR.RETRY         ; RETRYING PASS1.
                        ;36054
                        ;36055          ;1-----------------------------;
                        ;36056          R[TEMP5].SIZ_M[TEMP6].BCDSWP,   ; BEGIN TO REARRANGE BYTE ORDER IN TOYCN
U 0E7F, 0882,6637,0001,4047,00EA,C      ;36057          SIZE[BYTE]                      ; TEMP5_TOYCN<7:0,23:8,7:0>
                        ;36058
                        ;36059          ;--------------------------------;
U 0EAC, 0186,5C93,8037,F847,00EA,D      ;36060          M[TEMP5]_MB.ANDNOT.ZLIT24[0FF]  ; TEMP5_ZERO<31:24>'TOYCN<23:0>
                        ;36061
                        ;36062          ;--------------------------------;
J 0EAD, 0586,6C12,0037,F847,00EA,E      ;36063          M[TEMP6]_MB.AND.ZLIT0[0FF]      ; TEMP6_ZERO<31:8>'TOYCN<31:24>
```

```
                                ;36064          ;--------------------------------;
U 0EAE, 0880,5292,6001,8047,00EA,F ;36065       D_M[TEMP5].OR.(R[TEMP6].RL.24)  ; DREG_TOYCN
                                ;36066
                                ;36067          ;--------------------------------; DO FINAL SHIFT TO ALIGN TOYOS
                                ;36068          M[TEMP4]_(MB.RR.4).NOT,         ; OUTPUT OF TOYOS IS INVERTED.
U 0EAF, 0086,4276,0A31,0047,00E8,0 ;36069       WX.EQ.0?                        ; CHECK FOR BATTERY FAILURE
                                ;36070
                                ;36071 =0       ;0-------------------------------;
                                ;36072          M[TEMP2]_D+(R[TEMP4].RR.16),     ; RESULT IS TOYCN(DREG) + TOYOS(TEMP4)
U 0E80, 0886,22B1,0011,0047,00DA,6 ;36073       NEXT/MP.MFPR.CHK.CON            ;
                                ;36074
                                ;36075          ;1-------------------------------;
                                ;36076          M[TEMP2]_R[ZERO],               ; BATTERY FAILURE, RETURN ZERO
U 0E81, 0886,25BE,403D,8047,00DA,6 ;36077       NEXT/MP.MFPR.CHK.CON            ;
                                ;36078
                                ;36079 MP.MFPR.TU58:
                                ;36080          ;--------------------------------;
                                ;36081          M[TEMP2]_TU58REGS,              ;
                                ;36082          SL_[8],                         ;
U 0EB0, 0186,2F76,4030,43E7,00EB,2 ;36083       NEXT/MP.MFPR.CONSOLE.10         ;
                                ;36084
                                ;36085 MP.MFPR.CONSOLE:
                                ;36086          ;--------------------------------;
                                ;36087          M[TEMP2]_CONREGS,               ;
U 0EB1, 0986,2F76,4030,43C7,00EB,2 ;36088       SL_[8]                          ;
                                ;36089
                                ;36090 MP.MFPR.CONSOLE.10:
                                ;36091          ;--------------------------------;
U 0EB2, 0D80,0EF6,4030,8047,00E9,8 ;36092       PL_[16.],NEXT/MP.MFPR.XTR.WRITE ;
                                ;36093
                                ;36094 MP.MFPR.MEMSCR:
                                ;36095          ;--------------------------------;
                                ;36096          M[TEMP2]_MEMSCR,SL_[4],         ;
U 0EB3, 0586,2F76,4030,2647,00E9,8 ;36097       NEXT/MP.MFPR.XTR.WRITE          ;
```

```
                                ;36098 0E82:
                                ;36099 FREE.0E82:
                                ;36100          ;0------------------------------; FILLER NOT USED
U 0E82, 0986,BF37,0032,4047,017F,9 ;36101       M[ERRCOD]_OLIT16[72.],NEXT/IE.UNUSED.CS
                                ;36102
                                ;36103 0E83:
                                ;36104 FREE.0E83:
                                ;36105          ;1------------------------------; FPA IS NOT PRESENT
U 0E83, 0D86,BEB7,0031,C047,017F,9 ;36106       M[ERRCOD]_OLIT24[56.],NEXT/IE.UNUSED.CS
                                ;36107
                                ;36108 0FFE:
                                ;36109 MP.MFPR.ACCS:
                                ;36110          ;110******FORCE ADDRESS*********; FPA IS PRESENT
                                ;36111          M[TEMP2]_ZLIT0[1],              ;
U 0FFE, 0586,2C37,0030,0847,00DA,6 ;36112       NEXT/MP.MFPR.CHK.CON            ;
                                ;36113
                                ;36114 0FFF:    ;111******FORCE ADDRESS*********; FPA IS NOT PRESENT
                                ;36115          M[TEMP2]_ZLIT0[0],              ;
U 0FFF, 0186,2C37,0030,0047,00DA,6 ;36116       NEXT/MP.MFPR.CHK.CON            ;
                                ;36117
                                ;36118 MP.MFPR.MME:
                                ;36119          ;-------------------------------;
                                ;36120          M[TEMP2]_MEMSCR,SL_[1],         ;
U 0EB4, 0986,2F76,4030,0E47,00E9,8 ;36121       NEXT/MP.MFPR.XTR.WRITE          ;
                                ;36122
                                ;36123 MP.MFPR.TBDATA:
                                ;36124          ;-------------------------------;
U 0EB5, 0085,F592,4030,005F,00EB,6 ;36125       R[TEMP0]_TB                     ;
                                ;36126
                                ;36127          ;-------------------------------;
U 0EB6, 0C84,02F7,0030,0047,00EB,7 ;36128       R[TEMP0]_(M[TEMP0] RB).RR.9     ;
                                ;36129
                                ;36130          ;-------------------------------;
                                ;36131          M[TEMP2]_R[TEMP0],              ;
U 0EB7, 0886,25BE,4030,0047,00DA,6 ;36132       NEXT/MP.MFPR.CHK.CON            ;
                                ;36133
                                ;36134 MP.MFPR.PMR:
                                ;36135          ;-------------------------------;
U 0EB8, 0180,1C11,0032,E4A7,00EB,9 ;36136       VA_M[TEMP1]+ZLIT0[92.]          ; LOAD ADDRESS OF PMR BIT.
                                ;36137
                                ;36138          ;-------------------------------;
U 0EB9, 0186,2CB7,0034,0040,00EB,A ;36139       READ.PHY,M[TEMP2]_ZLIT24[80]    ; FETCH PMR AND LOAD MASK.
                                ;36140
                                ;36141          ;-------------------------------;
                                ;36142          R[TEMP2]_(M[MDR].AND.RB).RL.1,  ; MASK ONLY PMR AND ROTATE TO BIT 0.
U 0EBA, 0C85,2302,C030,8047,00DA,6 ;36143       NEXT/MP.MFPR.CHK.CON            ;
```

```
                                ;36144 MP.MFPR.SID:
                                ;36145          ;--------------------------------;
                                ;36146          M[TEMP1]_MB.RR.16,              ; ROTATE HARDWARE REV TO PROPER POSITION
U 0EBB, 10F6,13B7,0010,0047,00FF,6      ;36147          PATCH.NEXT/0FF6                 ; *** JUMP TO PCS ***
                                ;36148
                                ;36149 2BF6:
                                ;36150          ;================================; *** MICROWORD IN PCS ***
                                ;36151          M[TEMP2]_D.OR.ZLIT8[<MICROREV>],;
U 2BF6, 0186,2DB2,4033,1047,00FF,1      ;36152          NEXT/0FF7
                                ;36153
                                ;36154 0FF6:    ;*********FORCE ADDRESS*********;
                                ;36155          ;################################; *** MICROWORD REPLACED IN PCS ***
U 0FF6, 0D86,29B2,4032,F047,00FF,1      ;36156          M[TEMP2]_D.OR.ZLIT8[94.]        ;
                                ;36157
                                ;36158 0FF1:    ;*********FORCE ADDRESS*********;
                                ;36159          ;--------------------------------;
                                ;36160          R[TEMP2].SIZ_M[TEMP1],SIZE[BYTE], ;MERGE PROC ID, MICRO REV AND HARD REV
U 0FF1, 0C82,1592,4000,8047,00DA,6      ;36161          NEXT/MP.MFPR.CHK.CON            ;
                                ;36162
                                ;36163 =01
                                ;36164 MP.MFPR.DONE:
                                ;36165          ;01------------------------------; NOT IN CONSOLE MODE.
                                ;36166          Q_M[TEMP2],                     ; IL.MOV.B.W.L.MEM  DOES THE WRITE
U 0E59, 0080,2592,51B0,0047,0014,0      ;36167          LOD INC BRA?,NEXT/OS.WRT1       ; EVALUATE DESTINATION OPERAND
                                ;36168                                          ; OS.WRTI WILL BRANCH TO
                                ;36169                                          ; IL.MOV.B.W.L.MEM TO WRITE THE
                                ;36170                                          ; DESTINATION WITH THE VALUE IN Q.
                                ;36171
                                ;36172          ;11------------------------------; CONSOLE MODE.
                                ;36173          MDR_R[TEMP2],                   ; MOVE RESULT TO MDR FOR CONSOLE.
U 0E5B, 0080,05BE,4030,8467,0090,6      ;36174          NEXT/CN.E.GPR.IPR               ; RETURN TO CONSOLE MODE FLOWS.
                                ;36175
                                ;36176 =01
                                ;36177 MP.MFPR.BAD.IPR:
                                ;36178          ;01------------------------------; NOT IN CONSOLE MODE.
                                ;36179          CLEAR FLAG3,                    ; PUSH PCBACK-2
U 0E5D, 0418,0036,4030,0047,00FF,8      ;36180          NEXT/IE.OPER.FAULT              ; TAKE A RESERVED OPERAND FAULT.
                                ;36181
                                ;36182          ;11------------------------------; CONSOLE MODE.
                                ;36183          M[TEMP7]_ZLIT24[11],            ;
U 0E5F, 0D86,7CB7,0030,8847,0096,C      ;36184          NEXT/CN.ERROR                   ; RETURN TO CONSOLE MODE FLOWS.
```

```
                        ;36185 .TOC "  MTPR, MFPR, CHMX                  : CHMK, CHME, CHMS, CHMU"
                        ;36186
                        ;36187 ;*****************************************************************************
                        ;36188 ;       CHMX    code.rw
                        ;36189 ;
                        ;36190 ;       Input           MDR             Code operand
                        ;36191 ;
                        ;36192 ;       Resources       TEMP0           Hold PSL to extract CURM
                        ;36193 ;                       TEMP2           MINU ( TEMP7, PSL<CURM> )
                        ;36194 ;                       TEMP3           Save code operand
                        ;36195 ;                       TEMP4           Temp to switch stacks
                        ;36196 ;                       TEMP5           Save new PC
                        ;36197 ;                       TEMP7           Save mode selected by opcode
                        ;36198 ;                       MTEMP8          New PSL
                        ;36199 ;                       MDR
                        ;36200 ;                       SP
                        ;36201 ;                       KSP
                        ;36202 ;                       ESP
                        ;36203 ;                       SSP
                        ;36204 ;                       USP
                        ;36205 ;                       VA
                        ;36206 ;                       RNUM
                        ;36207 ;                       PL
                        ;36208 ;                       SL
                        ;36209 ;
                        ;36210 ;       Subroutines     PRB.WRITE1
                        ;36211 ;                       IE.PUSH.PSL.PC
                        ;36212 ;                       IE.LOAD.PC
                        ;36213 ;
                        ;36214 ;       CHMX CHANGES THE MODE TO THE SPECIFIED MODE UNLESS ALREADY RUNNING
                        ;36215 ;       AT A MORE PRIVELEDGED  MODE.  CHMX IS NOT ALLOWED IF ON THE INTERRUPT
                        ;36216 ;       STACK.  CHMX PROBES AHEAD TO MAKE SURE IT CAN PUSH 3 LONGWORDS ON
                        ;36217 ;       THE NEW STACK BEFORE INITIATING THE APPROPRIATE CHMX EXCEPTION.
                        ;36218 ;       IF THE STACK PROBE FAILS THE APPROPRIATE FAULT, ACV OR TNV,
                        ;36219 ;       IS INITIATED.
                        ;36220 ;
                        ;36221 ;       CHARLIE'S FLOWS 3.6
                        ;36222 ;*****************************************************************************
                        ;36223
                        ;36224 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;36225 CH.CHMK:
                        ;36226                 ;-------------------------------;
                        ;36227                 RTEMP7_RNUM_ZLIT0[0],           ; SAVE MODE SELECTED BY OPCODE.
U 03C6, 0185,DC37,0030,0047,00E6,3      ;36228                 NEXT/CA.CHMX.20                 ;
                        ;36229
                        ;36230 CH.CHME:
                        ;36231                 ;-------------------------------;
                        ;36232                 RTEMP7_RNUM_ZLIT0[1],           ; SAVE MODE SELECTED BY OPCODE.
U 03C7, 0D85,DC37,0030,0847,00EB,C      ;36233                 NEXT/CA.CHMX.10                 ;
                        ;36234
                        ;36235 CH.CHMS:
                        ;36236                 ;-------------------------------;
                        ;36237                 RTEMP7_RNUM_ZLIT0[2],           ; SAVE MODE SELECTED BY OPCODE.
U 03C8, 0D85,DC37,0030,1047,00EB,C      ;36238                 NEXT/CA.CHMX.10                 ;
```

```
                                        ;36239 CH.CHMU:
                                        ;36240          ;--------------------------------;
                                        ;36241          RTEMP7_RNUM_ZLIT0[3],           ; SAVE MODE SELECTED BY OPCODE.
U 03C9, 0985,DC37,0030,1847,00EB,C      ;36242          NEXT/CH.CHMX.10                 ;
                                        ;36243
                                        ;36244 .REGION/MPRCHM.R1L,MPRCHM.R1H/MPRCHM.R2L,MPRCHM.R2H/MPRCHM.R3L,MPRCHM.R3H
                                        ;36245
                                        ;36246 CH.CHMX.10:
                                        ;36247          ;--------------------------------; EXTRACT PSL<CURM> TO COMPARE WITH
U 0EBC, 0186,0EF6,4030,C087,00EB,D      ;36248          M[TEMP0]_PSL,PL_[24.]           ; MODE SELECTED BY OPCODE.
                                        ;36249
                                        ;36250          ;--------------------------------;
U 0EBD, 0580,0F76,4030,1047,00EB,E      ;36251          SL_[2]                          ;
                                        ;36252
                                        ;36253          ;--------------------------------;
                                        ;36254          WB_M[TEMP7]-(R[TEMP0].XZ),      ; COMPARE MODE SELECTED BY OPCODE
U 0EBE, 0860,7090,06E0,0047,08E6,1 400* ;36255          SIZE[LONG],WB<31-30>?           ; WITH PSL<CURM>.
                                        ;36256
                                        ;36257 =01      ;01-----------------------------;
                                        ;36258          R[TEMP2]_RNUM_MTEMP0.XZ,        ; NEW MODE IS PSL<CURM>.
U 0E61, 0085,D077,0030,8047,00EB,F      ;36259          NEXT/CH.CHMX.30                 ;
                                        ;36260
                                        ;36261 CH.CHMX.20:
                                        ;36262          ;11-----------------------------;
U 0E63, 0484,7592,4030,8047,00EB,F      ;36263          R[TEMP2]_M[TEMP7]               ; NEW MODE IS MODE SELECTED BY OPCODE
                                        ;36264
                                        ;36265 CH.CHMX.30:
                                        ;36266          ;--------------------------------;
                                        ;36267          M[TEMP4]_R[SP],                 ; PREPARE TO SWITCH STACKS.
U 0EBF, 0886,45BE,47B7,8207,08E0,8 479* ;36268          PSL<IS.CURM>?                   ;
                                        ;36269
                                        ;36270 =1000    ;1000---------------------------;
                                        ;36271          R[KSP]_M[TEMP4],                ; SAVE SP IN KSP.
U 0E08, 0C84,4592,4038,0047,00EC,0      ;36272          NEXT/CH.CHMX.40                 ;
                                        ;36273
                                        ;36274          ;1001---------------------------;
                                        ;36275          R[ESP]_M[TEMP4],                ; SAVE SP IN ESP.
U 0E09, 0884,4592,4038,4047,00EC,0      ;36276          NEXT/CH.CHMX.40                 ;
                                        ;36277
                                        ;36278          ;1010---------------------------;
                                        ;36279          R[SSP]_M[TEMP4],                ; SAVE SP IN SSP.
U 0E0A, 0884,4592,4038,8047,00EC,0      ;36280          NEXT/CH.CHMX.40                 ;
                                        ;36281
                                        ;36282          ;1011---------------------------;
                                        ;36283          R[USP]_M[TEMP4],                ; SAVE SP IN USP.
U 0E0B, 0C84,4592,4038,C047,00EC,0      ;36284          NEXT/CH.CHMX.40                 ;
                                        ;36285
                                        ;36286          ;1100---------------------------; CHMX IS UNDEFINED ON THE INT STACK
                                        ;36287          M[FPDOFFSET]_ZLIT0[0A],         ; SET HALT ERROR CODE
                                        ;36288          SET STACK FLAG,                 ; TELL CONSOLE TO LOOK AT FRONT PANEL
U 0E0C, 056E,CC37,0030,5047,0000,1      ;36289          NEXT/MS.HALT.MICRO              ; HALT.
                                        ;36290 =
                                        ;36291 CH.CHMX.40:
                                        ;36292          ;--------------------------------; PREPARE TO PROBE NEW MODE STACK AREA
U 0EC0, 0486,4E7D,003C,84A7,00EC,1      ;36293          VA_M[TEMP4]_R[IPR.R]-1          ; FROM NEW SP-1 TO NEW SP-12.
```

```
                        ;36294          ;--------------------------------;
                        ;36295          R[TEMP3]_SEXT(M[MDR]),          ; SAVE OPERAND IN TEMP3.
U 0EC1, 0085,2E5E,0010,C047,00E6,4      ;36296          SIZE[WORD]              ;
                        ;36297
                        ;36298 =00      ;00-------------------------------;  *** MM.PRB.WRITE1 DESTROYS MDR ***
                        ;36299          M[TEMP2]_MB.RL.24,              ; PROBE SP-1 WITH THE MODE
U 0E64, 0486,23B7,0000,0047,0579,B      ;36300          PUSH,NEXT/MM.PRB.WRITE1         ; SPECIFIED IN TEMP2 (RETURN IF OK).
                        ;36301
                        ;36302          ;01-------------------------------;  *** MM.PRB.WRITE1 DESTROYS MDR ***
                        ;36303          VA_M[TEMP4]-ZLIT0[11.],         ; PROBE SP-12 WITH THE MODE
U 0E65, 0D80,4C10,0030,5CA7,0579,B      ;36304          PUSH,NEXT/MM.PRB.WRITE1         ; SPECIFIED IN TEMP2 (RETURN IF OK).
                        ;36305
                        ;36306          ;10-------------------------------;  CALCULATE MACRO VECTOR ADDRESS
                        ;36307          M[TEMP7]_(MB+R[TEMP7]),0L.1,    ; OFFSET = TEMP7 * 4
U 0E66, 0C66,7301,C031,C047,00EC,2      ;36308          SET MM.NOINT            ;
                        ;36309 =
                        ;36310          ;--------------------------------;
                        ;36311          R[TEMP7]_M[SCBB]+RB,            ; ADD OFFSET TO SCBB.
U 0EC2, 0C84,E001,0B31,D8E7,00E8,4      ;36312          IP.TS?                  ; CHECK FOR INTERRUTPS.
                        ;36313
                        ;36314 =0       ;0-------------------------------;  NO INTERRUPT, OR RETURN FROM TIMER
                        ;36315          VA_M[TEMP7]+ZLIT0[40],          ; SERVICE, LOAD MACRO VECTOR ADDRESS.
U 0E84, 0180,7C11,0032,04A7,00EC,3      ;36316          NEXT/CH.CHMX.45         ;
                        ;36317
                        ;36318          ;1-------------------------------;
                        ;36319          INTPEND OR TIMER?,              ; SERVICE TIMER OR INTERRUPT
U 0E85, C480,0036,4AF0,0047,04F7,0      ;36320          PUSH,NEXT/IE.SERV.IP.TS2        ;
                        ;36321
                        ;36322 CH.CHMX.45:
                        ;36323          ;--------------------------------;
U 0EC3, 0C80,0036,4030,0040,00EC,4      ;36324          READ.PHY                ; READ MACRO VECTOR
                        ;36325
                        ;36326          ;--------------------------------;
                        ;36327          R[TEMP5]_M[MDR],CLEAR FLAG2,    ; SAVE MACRO VECTOR IN TEMP5.
U 0EC4, 0015,2592,43B1,4047,00E8,6      ;36328          WB<1-0>.NE.0?           ; BRANCH ON MACRO VECTOR<1:0>=0.
                        ;36329
                        ;36330 =0       ;0-------------------------------;  MACRO VECTOR<1:0>=0,
                        ;36331          R[SP]_M[TEMP4]+1,               ; LOAD NEW SP AND GO
U 0E86, 0C5C,4E50,0037,8047,00EC,5      ;36332          SET FLAG3,NEXT/CH.CHMX.50       ; PUSH PC, NOT PCBACK-2
                        ;36333
                        ;36334          ;1-------------------------------;  MACRO VECTOR<1:0> NOT EQUAL TO 0,
                        ;36335          M[FPDOFFSET]_ZLIT0[0B],         ; OPERATION UNDEFINED. SET HALT CODE
                        ;36336          SET STACK FLAG,                 ; TELL CONSOLE TO LOOK AT FRONT PANEL
U 0E87, 016E,CC37,0030,5847,0000,1      ;36337          NEXT/MS.HALT.MICRO      ; AND HALT.
                        ;36338
                        ;36339 CH.CHMX.50:
                        ;36340          ;--------------------------------;
                        ;36341          VA_R[SP]_RB-CONX(4),            ;
U 0EC5, 0C6C,073C,0027,84A7,00EC,6      ;36342          SET STACK FLAG          ; PREPARE TO PUSH ONTO THE STACK.
                        ;36343
                        ;36344          ;--------------------------------;
U 0EC6, 0086,4036,4030,0087,00EC,7      ;36345          M[TEMP4]_PSL            ; SAVE OLD PSL TO PUSH ON STACK.
                        ;36346
                        ;36347          ;--------------------------------;
U 0EC7, 0880,05BE,4030,86A7,00EC,8      ;36348          PSL(PREV_CURM ISCURM_R[TEMP2])  ; BEGIN TO BUILD NEW PSL.
```

```
                                ;36349          ;--------------------------------;
U 0EC8, 0886,8036,4030,0087,00E8,8      ;36350          M[TEMP8]_PSL                    ; MOVE NEW PSL INTO MTEMP8.
                                ;36351
                                ;36352 =0       ;0-------------------------------;
                                ;36353          M[TEMP8]_MB.AND.OLIT24[107],    ; CLEAR NEW PSL<CM,TP,FPD>
U 0E88, 0986,8E92,0038,3847,04F0,5      ;36354          PUSH,NEXT/IE.105                ; GO PUSH PSL AND PC ON THE STACK.
                                ;36355
                                ;36356          ;1-------------------------------;
U 0E89, 0C84,073C,0027,84A7,00EC,9      ;36357          VA_R[SP]_RB-CONX(4)             ; PREPARE TO PUSH OPERAND ON THE STACK.
                                ;36358
                                ;36359          ;--------------------------------;
                                ;36360          WRITE M[TEMP3],SIZE[LONG],      ; PUSH OPERAND ON THE STACK.
U 0EC9, 0C80,3592,4020,05D8,00EF,5      ;36361          NEXT/IE.LOAD.PC01               ; FINISH INITIATING THE EXCEPTION.
```

```
                                        ;36362 .TOC  "      MTPR, MFPR, CHMX                : Ird Rom Definition"
;36363  .NOBIN
;36364
;36365  .ICODE  ;       OPS             REG                     MEM                OPS         FPA REG                 FPA MEM
;36366       OBD: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                    ;CHME
;36367            IRD1[LOD][OS.RED                ]                                [LOD][OS.RED                ]
;36368  .OCODE
;36369       OBD: CNTO[NOP][CH.CHME               ][CH.CHME               ]  [NOP][CH.CHME               ][CH.CHME               ],
;36370            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;36371
;36372  .ICODE
;36373       OBC: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                    ;CHMK
;36374            IRD1[LOD][OS.RED                ]                                [LOD][OS.RED                ]
;36375  .OCODE
;36376       OBC: CNTO[NOP][CH.CHMK               ][CH.CHMK               ]  [NOP][CH.CHMK               ][CH.CHMK               ],
;36377            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;36378
;36379  .ICODE
;36380       OBE: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                    ;CHMS
;36381            IRD1[LOD][OS.RED                ]                                [LOD][OS.RED                ]
;36382  .OCODE
;36383       OBE: CNTO[NOP][CH.CHMS               ][CH.CHMS               ]  [NOP][CH.CHMS               ][CH.CHMS               ],
;36384            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;36385
;36386  .ICODE
;36387       OBF: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                    ;CHMU
;36388            IRD1[LOD][OS.RED                ]                                [LOD][OS.RED                ]
;36389  .OCODE
;36390       OBF: CNTO[NOP][CH.CHMU               ][CH.CHMU               ]  [NOP][CH.CHMU               ][CH.CHMU               ],
;36391            CNT1[NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]  [NOP][IE.BAD.IRD            ][IE.BAD.IRD            ]
;36392
;36393  .ICODE
;36394       ODB: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                    ;MFPR
;36395            IRD1[LOD][OS.RED                ]                                [LOD][OS.RED                ]
;36396  .OCODE
;36397       ODB: CNTO[NOP][MP.MFPR               ][MP.MFPR               ]  [NOP][MP.MFPR.FPA           ][MP.MFPR.FPA           ],
;36398            CNT1[NOP][IL.MOV.B.W.L.MEM      ][IL.MOV.B.W.L.MEM      ]  [NOP][IL.MOV.B.W.L.MEM      ][IL.MOV.B.W.L.MEM      ]
;36399
;36400  .ICODE
;36401       ODA: FPD [NOP][IE.OPCOD.DEC          ]                                [NOP][IE.OPCOD.DEC          ],                    ;MTPR
;36402            IRD1[LOD][OS.RED                ]                                [LOD][OS.RED                ]
;36403  .OCODE
;36404       ODA: CNTO[LOD][OS.RED                ][OS.RED                ]  [LOD][OS.RED                ][OS.RED                ],
;36405            CNT1[NOP][MP.MTPR               ][MP.MTPR               ]  [NOP][MP.MTPR               ][MP.MTPR               ]
```

```
;36406  .TOC    ''       MTPR, MFPR, CHMX                 : Dsize Rom Definition''
;36407  .DCODE
;36408
;36409  OBD:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;CHME
;36410
;36411  OBC:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;CHMK
;36412
;36413  OBE:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;CHMS
;36414
;36415  OBF:    SIZE [WORD] [   0] [   0] [   0] [   0] [   0]  ;CHMU
;36416
;36417  ODB:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;MFPR
;36418
;36419  ODA:    SIZE [LONG] [LONG] [   0] [   0] [   0] [   0]  ;MTPR
;36420
;36421  .UCODE
                                        ;36422 .BIN
```

36422; This page intentionally left blank.

```
                                        ;36423 .TOC "PRLDSV.MIC"
                                        ;36424 .TOC "REVISION 19.0"
                                        ;36425 ;       CHARLIE MCDOWELL
                                        ;36426
;36427  .NOBIN
;36428
;36429  .TOC "  Revision History"
;36430
;36431  ; REV   EXPLAINATION
;36432
;36433  ; 19    Fix PROBE instructions to flush the XB after restoring the PSL to
;36434  ;       avoid erroneous ACVs due to prefetch while PSL had modified current
;36435  ;       mode bits to execute the probe.  Also fix to report no access if
;36436  ;       probe across a page boundary with first page access ok but not valid,
;36437  ;       and second page no access.
;36438  ; 18    29-MAY-1980
;36439  ;       Change SVPCTX to save the SP in KSP as well as the PCB if executed
;36440  ;       while on the kernal stack.
;36441  ; 17    Add BUS/PRB.RD.PTE to avoid machine hang clearing TB.
;36442  ; 16    Initial release.
                                        ;36443         .BIN
```

```
                                ;36444 .TOC "  PROBE, LDPCTX, SVPCTX           : PROBER"
                                ;36445
                                ;36446 ;******************************************************************************
                                ;36447 ;       PROBER  mode.rb, len.rw, base.ab
                                ;36448 ;
                                ;36449 ;       Input           Q               Mode
                                ;36450 ;                       MDR             Length
                                ;36451 ;
                                ;36452 ;       Resources       TEMP1           Save PSL
                                ;36453 ;                       TEMP3           Save ZEXT(Length) - 1
                                ;36454 ;                       MM.TEMP1        Save VA if TB Miss
                                ;36455 ;
                                ;36456 ;       Subroutines     PR.PROBEX.SUB
                                ;36457 ;
                                ;36458 ;       CHARLIE'S FLOWS 3.1
                                ;36459 ;******************************************************************************
                                ;36460
                                ;36461 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;36462 =0
                                ;36463 PR.PROBER:
                                ;36464         ;0-----------------------------;
                                ;36465         R[TEMP3]_ZEXT(M[MDR]),SIZE[WORD]; SAVE ZERO EXTENDED LENGTH
U 032A, 0885,259E,4010,C047,0573,9  ;36466     PUSH,NEXT/PR.PROBEX.SUB         ;
                                ;36467
                                ;36468         ,!----------------------------;
                                ;36469         R[TEMP3]_M[TEMP3]-1,            ; SAVE ZEXT(LENGTH) - 1
U 032B, 0484,3E51,0780,C05F,094E,C 479*  ;36470     SIZE[BYTE],PROBE READ?      ;
                                ;36471 .REGION/PRLDSV.R1L,PRLDSV.R1H/PRLDSV.R2L,PRLDSV.R2H/PRLDSV.R3L,PRLDSV.R3H
                                ;36472
                                ;36473 =1100    ;1100----------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                ;36474         R[MM.TEMP1]_M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 14EC, 0485,6592,4033,C047,0562,8  ;36475     PUSH,NEXT/MM.PRB.READ.TBM       ; LOAD THE PTE BACK IN THE TB.
                                ;36476
                                ;36477         ;1101----------------------------; CATCH RETURN+1 FROM MM.PRB.READ.TBM
                                ;36478         WB_M[ERRCOD].AND.ZLIT0[2],      ; CHECK IF FAULT REFERENCEING SPTE
U 14ED, 0180,BC12,0A30,1047,0557,C  ;36479     PUSH,WX.EQ.0?,NEXT/PR.TNV       ;
                                ;36480
                                ;36481         ;1110----------------------------; PROBE NO ACCESS OR RETURN+2 FROM
                                ;36482         PSL_M[TEMP1],                   ; MM.PRB.READ.TBM, RESTORE PSL
U 14EE, 0C80,1002,403D,8007,016C,D  ;36483     NEXT/PR.PROBEX.ACV              ; SET CONDITION CODES AND DO IRD1
                                ;36484
                                ;36485         ;1111----------------------------; READ ACCESS OK OR RETURN+3 FROM
U 14EF, 0C81,B001,0030,C4A7,016B,7  ;36486     VA_M[VA]+R[TEMP3]               ; MM.PRB.READ.TBM, CONTINUE.
```

```
                        ;36487          ;--------------------------------;
U 16B7, 0C80,0036,4780,005F,0954,C 479* ;36488          $IZE[BYTE],PROBE READ?           ;
                        ;36489
                        ;36490 =1100    ;1100----------------------------; PROBE NOT VALID (PTE NOT IN TB)
                        ;36491          R[MM.TEMP1] M[VA],               ; SAVE VA AND PROCESS A TB MISS TO
U 154C, 0485,B592,4033,C047,0562,8      ;36492          PUSH.NEXT/MM.PRB.READ.TBM        ; LOAD THE PTE BACK IN THE TB.
                        ;36493
                        ;36494          ;1101----------------------------; CATCH RETURN+1 FROM MM.PRB.READ.TBM
                        ;36495          WB_M[ERRCOD].AND.ZLITO[2],       ; CHECK IF FAULT REFERENCING SPTE
U 154D, 0180,BC12,0A30,1047,0557,C      ;36496          PUSH,WX.EQ.0?,NEXT/PR.TNV        ;
                        ;36497
                        ;36498          ;1110----------------------------; PROBE NO ACCESS OR RETURN+2 FROM
                        ;36499          PSL_M[TEMP1],                    ; MM.PRB.READ.TBM, RESTORE PSL
U 154E, 0C80,1002,403D,8007,016C,D      ;36500          NEXT/PR.PROBEX.ACV               ; SET CONDITION CODES AND DO IRD1
                        ;36501
                        ;36502          ;1 11----------------------------; READ ACCESS OK OR RETURN+3 FROM
                        ;36503          PSL_M[TEMP1],                    ; MM.PRB.READ.TBM, RESTORE PSL.
U 154F, 0480,1002,403D,8007,017F,8      ;36504          NEXT/PR.PROBEX.OK                ; SET CC TO INDICATE ACCESS OK.
                        ;36505 =0
                        ;36506 PR.TNV:
                        ;36507          ;0-----------------------------; FAULT FETCHING SPTE
                        ;36508          PSL_M[TEMP1],                    ; RESTORE PSL AND
U 157C, 0480,1002,403D,8007,00F6,1      ;36509          NEXT/IE.TNV                      ; TAKE A TNV FAULT.
                        ;36510
                        ;36511          ;1-----------------------------; NOT A FAULT FETCHING SPTE
U 157D, 0880,0036,40B0,0047,0000,2      ;36512          RETURN [+2]                      ; RETURN TO INDICATE ACCESS OK.
                        ;36513
                        ;36514 17F8:    ;*********FORCE ADDRESS*********;
                        ;36515 PR.PROBEX.OK:
                        ;36516          ;--------------------------------;
                        ;36517          WB_R[ZERO]+1,CCOP2,SIZE[IDEP],   ; SET CC TO INDICATE ACCESS OK.
U 17F8, 0C80,0E7C,003D,9047,016E,E      ;36518          NEXT/PR.PROBEX.FLUSH             ;
                        ;36519
                        ;36520 PR.PROBEX.ACV:
                        ;36521          ;--------------------------------;
                        ;36522          WB_R[ZERO],CCOP2,SIZE[IDEP],     ; SET CC TO INDICATE NO ACCESS.
U 16CD, 0880,05BE,403D,9047,016E,E      ;36523          NEXT/PR.PROBEX.FLUSH             ;
                        ;36524
                        ;36525 PR.PROBEX.FLUSH:
                        ;36526          ;--------------------------------; FLUSH XB TO AVOID ERRONEOUS ACV DUE
U 16EE, 0481,A592,4030,0487,0000,2      ;36527          FLUSH XB,NEXT/GL.NOP.IRD1        ; TO PREFETCH WHILE PSL WAS CHANGED
```

```
                                ;36528 .TOC "  PROBE, LDPCTX, SVPCTX          : PROBEW"
                                ;36529
                                ;36530 ;*****************************************************************************
                                ;36531 ;       PROBEW  mode.rb, len.rw, base.ab
                                ;36532 ;
                                ;36533 ;       Input           Q               Mode
                                ;36534 ;                       MDR             Length
                                ;36535 ;
                                ;36536 ;       Resources       TEMP1           Save PSL
                                ;36537 ;                       TEMP3           Save ZEXT(Length) - 1
                                ;36538 ;                       MM.TEMP1        Save VA if TB Miss
                                ;36539 ;
                                ;36540 ;       Subroutines     PR.PROBEX.SUB
                                ;36541 ;
                                ;36542 ;        CHARLIE'S FLOWS 3.1
                                ;36543 ;*****************************************************************************
                                ;36544
                                ;36545 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;36546 =0
                                ;36547 PR.PROBEW:
                                ;36548         ;0------------------------------;
                                ;36549         R[TEMP3]_ZEXT(M[MDR]),SIZE[WORD],; SAVE ZERO EXTENDED LENGTH
U 0332, 0885,259E,4010,C047,0573,9 ;36550      PUSH,NEXT/PR.PROBEX.SUB         ;
                                ;36551
                                ;36552         ;1------------------------------;
                                ;36553         R[TEMP3]_M[TEMP3]-1,            ; SAVE ZEXT(LENGTH) - 1
U 0333, 0C84,3E51,0780,C05D,0955,C 479* ;36554 SIZE[BYTE],PROBE_WRITE?          ;
                                ;36555 .REGION/PRLDSV.R1L,PRLDSV.R1H/PRLDSV.R2L,PRLDSV.R2H/PRLDSV.R3L,PRLDSV.R3H
                                ;36556
                                ;36557 =1100    ;1100------------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                ;36558         R[MM.TEMP1]_M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 155C, 0485,B592,4033,C047,0561,8 ;36559      PUSH,NEXT/MM.PRB.WRT.TBM        ; LOAD THE PTE BACK IN THE TB.
                                ;36560
                                ;36561         ;1101------------------------------; CATCH RETURN+1 FROM MM.PRB.WRT.TBM
                                ;36562         WB_M[ERRCOD].AND.2LITO[2],      ; CHECK IF FAULT REFERENCING SPTE
U 155D, 0180,BC12,0A30,1047,0557,C ;36563      PUSH,WX.EQ.0?,NEXT/PR.TNV       ;
                                ;36564
                                ;36565         ;1110------------------------------; PROBE NO ACCESS OR RETURN+2 FROM
                                ;36566         PSL_M[TEMP1],                   ; MM.PRB.WRT.TBM, RESTORE PSL
U 155E, 0C80,1002,403D,8007,016C,D ;36567      NEXT/PR.PROBEX.ACV              ; SET CONDITION CODES AND DO IRD1
                                ;36568
                                ;36569         ;1111------------------------------; WRITE ACCESS OK OR RETURN+3 FROM
U 155F, 0C81,8001,0030,C4A7,0173,8 ;36570      VA_M[VA]+R[TEMP3]               ; MM.PRB.WRT.TBM, CONTINUE.
```

```
                                    ;36571          ;--------------------------------;
U 1738, 0C80,0036,4780,005D,0956,C 479*  ;36572     SIZE[BYTE],PROBE WRITE?          ;
                                    ;36573
                                    ;36574 =1100    ;1100-----------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                    ;36575          R[MM.TEMP1] M[VA],               ; SAVE VA AND PROCESS A TB MISS TO
U 156C, 0485,B592,4033,C047,0561,8  ;36576          PUSH,NEXT/MM.PRB.WRT.TBM         ; LOAD THE PTE BACK IN THE TB.
                                    ;36577
                                    ;36578          ;1101-----------------------------; CATCH RETURN+1 FROM MM.PRB.WRT.TBM
                                    ;36579          WB_M[ERRCOD].AND.ZLIT0[2],       ; CHECK IF FAULT REFERENCING SPTE
U 156D, 0180,BC12,0A30,1047,0557,C  ;36580          PUSH,WX.EQ.0?,NEXT/PR.TNV        ;
                                    ;36581
                                    ;36582          ;1110-----------------------------; PROBE NO ACCESS OR RETURN+2 FROM
                                    ;36583          PSL_M[TEMP1],                    ; MM.PRB.WRT.TBM, RESTORE PSL
U 156E, 0C80,1002,403D,8007,016C,D  ;36584          NEXT/PR.PROBEX.ACV               ; SET CONDITION CODES AND DO IRD1
                                    ;36585
                                    ;36586          ;1111-----------------------------; WRITE ACCESS OK OR RETURN+3 FROM
                                    ;36587          PSL_M[TEMP1],                    ; MM.PRB.WRT.TBM, RESTORE PSL.
U 156F, 0480,1002,403D,8007,017F,8  ;36588          NEXT/PR.PROBEX.OK                ;
```

```
                        ;36589 .TOC "  PROBE, LDPCTX, SVPCTX           : PR.PROBEX.SUB"
                        ;36590
                        ;36591 ;*****************************************************************************
                        ;36592 ;       Entry points    PR.PROBEX.SUB
                        ;36593 ;
                        ;36594 ;       Input           Q               Probe mode
                        ;36595 ;
                        ;36596 ;       Output          PSL<CURM>       MAX( PSL<PREV>, Probe mode)
                        ;36597 ;                       TEMP1           Save PSL
                        ;36598 ;
                        ;36599 ;       Resources       TEMP2           Extracted PSL<PREV>
                        ;36600 ;                       TEMP7           Temporary save specified mode
                        ;36601 ;                       PL              Used for loading PSL<CURM>
                        ;36602 ;                       SL              Used for extracting mode bits
                        ;36603 ;
                        ;36604 ;       Subroutines     OS.ADD
                        ;36605 ;
                        ;36606 ;       PROBEX.SUB IS USED BY PROBER AND PROBEW TO PROCESS THE OPERANDS
                        ;36607 ;       AND PREPARE TO PROBE.
                        ;36608 ;
                        ;36609 ;       CHARLIE'S FLOWS 3.1.2
                        ;36610 ;*****************************************************************************
                        ;36611
                        ;36612 PR.PROBEX.SUB:
                        ;36613          ;-------------------------------;
                        ;36614          SET_MM.NOINT,                   ;
U 1739, 0560,0F76,4030,1047,0173,C      ;36615          SL_[2]                          ; USED TO EXTRACT MODE WHICH IS 2 BITS.
                        ;36616
                        ;36617          ;-------------------------------;
                        ;36618          M[TEMP1]_PSL,                   ; SAVE PSL.
U 173C, 0D86,1EF6,4030,B087,0173,D      ;36619          PL_[22.]                        ; POSITION OF PREV MODE IN PSL.
                        ;36620
                        ;36621          ;-------------------------------;
U 173D, 0484,1077,0030,8047,0158,C      ;36622          R[TEMP2]_M[TEMP1].XZ            ; GET PSL<PREV>.
                        ;36623
                        ;36624 =0       ;0------------------------------;
                        ;36625          M[TEMP7]_Q Q_D,                 ; SAVE MODE OPERAND.
                        ;36626          PL_[0],                         ; POSITION OF MODE OPERAND MODE BITS.
U 158C, 0186,7EEC,71B0,0047,0412,0      ;36627          PUSH,LOD INC BRA?,NEXT/OS.ADD   ; FETCH BASE OPERAND.
                        ;36628
                        ;36629          ;1------------------------------;
                        ;36630          WB_M[TEMP2]-(R[TEMP7].XZ),      ; COMPARE PSL<PREV MODE> WITH
U 158D, 0080,2090,06C1,C047,0959,9 400* ;36631          SIZE[BYTE],WB<31-30>?           ; SPECIFIED MODE.
                        ;36632
                        ;36633 =01
                        ;36634          ;01-----------------------------;
                        ;36635          PSL<CURM>_M[TEMP2].RR.8,        ; USE PSL<PREV MODE>
U 1599, 0480,23B7,0000,06A7,0173,E      ;36636          NEXT/PR.PROBEX.SUB.10           ;
                        ;36637
                        ;36638          ;11-----------------------------;
U 159B, 0480,73B7,0000,06A7,0173,E      ;36639          PSL<CURM>_M[TEMP7].RR.8         ; USE MODE SPECIFIED IN OPERAND
                        ;36640
                        ;36641 PR.PROBEX.SUB.10:
                        ;36642          ;-------------------------------;
U 173E, 0C81,2002,40BD,84A7,0000,1      ;36643          VA_M[MDR],RETURN [+1]           ; MOVE BASE ADDRESS INTO VA
```

```
                                ;36644 .TOC "  PROBE, LDPCTX, SVPCTX          : LDPCTX"
                                ;36645
                                ;36646 ;*****************************************************************************
                                ;36647 ;       LDPCTX          Load Process Context
                                ;36648 ;
                                ;36649 ;       Input           PCBB            Process Control Block pointed to by PCBB
                                ;36650 ;
                                ;36651 ;       Resources       TEMP0           Hold PCBB on MBUS
                                ;36652 ;                       TEMP1           Temporary
                                ;36653 ;                       FLAG1           Used to repeat Clear TB loop once
                                ;36654 ;                       STEPC           Clear TB loop count
                                ;36655 ;                       Q               Used to load RNUM for loading GPR'S
                                ;36656 ;
                                ;36657 ;       Subroutines     LS.MODE.CHECK
                                ;36658 ;                       IE.SERV.IP.TS
                                ;36659 ;
                                ;36660 ;       CHARLIE'S FLOWS 3.7
                                ;36661 ;*****************************************************************************
                                ;36662
                                ;36663 .REGION/IRD1.R1L,IRD1.R1H
                                ;36664 =000
                                ;36665 LS.LDPCTX:
                                ;36666          ;000-------------------------------;
U 03A0, 0980,0C37,2030,04A7,015A,8  ;36667      VA_D_ZLIT0[0]                   ; PREPARE TO INVALIDATE PROCESS TB
                                ;36668 =
                                ;36669
                                ;36670 .REGION/PRLDSV.R1L,PRLDSV.R1H/PRLDSV.R2L,PRLDSV.R2H/PRLDSV.R3L,PRLDSV.R3H
                                ;36671
                                ;36672 =00      ;00-------------------------------;
                                ;36673          Q -1,                           ; PREPARE TO LOAD GPR'S
                                ;36674          PSL<IS.CURM>?,                  ; CHECK FOR PRIVILEGED INSTRUCTION FAULT
U 15A8, 0880,0E77,1730,0207,0D60,C 479*  ;36675 PUSH,NEXT/LS.MODE.CHECK          ;
                                ;36676
                                ;36677          ;01-------------------------------; INVALIDATE PROCESS SPACE TB ENTRIES
                                ;36678          TB_D+ZLIT8[4],                  ; INITIALIZE ALU, MUX, ROT, AND LITRL
U 15A9, 0581,BDB1,0030,2507,04E5,4  ;36679      PUSH,NEXT/MP.INVALIDATE.TB      ; THESE FIELDS ARE THE SAME IN THE LOOP
                                ;36680
                                ;36681          ;10-------------------------------;
                                ;36682          STEPC_14.,                      ;
                                ;36683          VA_M[TEMP0]+ZLIT0[16.],         ; LOAD ADDRESS OF R0 IN PCB.
U 15AA, 0DB0,0C11,0B20,84A7,0959,6 416*  ;36684 SIZE[LONG],ALLOW INT?           ; ('LSS 0' MUST BE 0, SEE CCBR CHART)
                                ;36685 =
                                ;36686 =0       ;0--------------------------------; CATCH RETURN-1 FORM IE.SERV.IP.TS
                                ;36687          READ.PHY,RNUM_Q_Q+1,            ; FETCH GPR FROM PCB.
U 1596, 0881,D0B9,103D,8040,0173,F  ;36688      NEXT/LS.LDPCTX.GPRS.1           ;
                                ;36689
                                ;36690          ;1-------------------------------; SERVICE PENDING INTERRUPT OR
                                ;36691          INTPEND OR TIMER?,              ;
U 1597, 0480,0036,4AF0,0047,04F7,0  ;36692      PUSH,NEXT/IE.SERV.IP.TS2        ; TIMER SERVICE
```

```
                                ;36693 =0
                                ;36694 LS.LDPCTX.GPRS.0:
                                ;36695          ;0-------------------------------;
                                ;36696          READ.PHY,RNUM_Q_Q+1,            ; FETCH GPR FROM PCB.
U 159C, 0881,D0B9,103D,8040,0173,F ;36697       NEXT/LS.LDPCTX.GPRS.1           ;
                                ;36698
                                ;36699          ;1-------------------------------;
                                ;36700          VA_M[TEMP0]+ZLIT0[80.],         ; LOAD ADDRESS OF P0BR IN PCB.
U 159D, 0180,0C11,0032,84A7,0174,0 ;36701       NEXT/LS.LDPCTX.P0BR             ;
                                ;36702
                                ;36703 LS.LDPCTX.GPRS.1:
                                ;36704          ;-------------------------------;
                                ;36705          R[GPR.R]_M[MDR],VA_VA+4,        ; LOAD GPR(RNUM).
                                ;36706          DBZ STEPC?,                     ;
U 173F, 0485,2592,433C,C447,0159,C ;36707       NEXT/LS.LDPCTX.GPRS.0           ;
                                ;36708
                                ;36709 LS.LDPCTX.P0BR:
                                ;36710          ;-------------------------------;
                                ;36711          READ.PHY,VA_VA+4,               ; FETCH P0BR FROM PCB.
U 1740, 0B80,0039,FFFF,FC40,0174,1 ;36712       LONLIT_[0F8C00000]              ; LOAD MASK FOR CHECKING MBZ BITS.
                                ;36713
                                ;36714          ;-------------------------------;
U 1741, 0581,AC10,0030,0C87,0174,2 ;36715       PC_M[PC]-ZLIT0[1]               ; CORRECT PC FOR NO OPERANDS.
                                ;36716
                                ;36717          ;-------------------------------;
                                ;36718          R[P0BR]_M[MDR],                 ; LOAD P0BR.
U 1742, 0C85,2592,403A,0440,0174,3 ;36719       READ.PHY,VA_VA+4                ; FETCH P0LR AND ASTLVL FROM PCB.
                                ;36720
                                ;36721          ;-------------------------------;
                                ;36722          WB_M[MDR].AND.R[LONLIT],        ; CHECK MBZ BITS.
U 1743, 0C81,2002,0A3D,4047,015A,6 ;36723       WX.EQ.0?                        ;
                                ;36724
                                ;36725 =0       ;0-------------------------------;
U 15A4, 0C80,0036,4030,0047,00FF,8 ;36726       NEXT/IE.OPER.FAULT              ; RESERVED OPERAND FAULT.
                                ;36727
                                ;36728          ;1-------------------------------;
U 15A7, 0085,2592,4030,4707,0174,4 ;36729       ASTLVL_R[TEMP1]_M[MDR]  ; LOAD ASTLVL.
                                ;36730
                                ;36731          ;-------------------------------;
U 1744, 0986,1C93,8030,3847,0174,5 ;36732       M[TEMP1]_MB.ANDNOT.ZLIT24[7]    ;
                                ;36733
                                ;36734          ;-------------------------------;
                                ;36735          READ.PHY,VA_VA+4,               ; FETCH P1BR FROM PCB.
U 1745, 0C84,1592,403A,4440,0174,6 ;36736       R[P0LR]_M[TEMP1]                ; LOAD P0LR.
```

```
                                ;36737          ;--------------------------------;
                                ;36738          READ.PHY,                        ; FETCH P1LR FROM PCB.
U 1746, 0885,2592,403A,8040,0174,7 ;36739       R[P1BR]_M[MDR]                   ; LOAD P1BR.
                                ;36740
                                ;36741          ;--------------------------------;
U 1747, 0B80,0401,FFFF,F847,0174,8 ;36742       LONLIT_[7FC00000]                ; LOAD MASK FOR MBZ CHECK.
                                ;36743
                                ;36744          ;--------------------------------;
U 1748, 0980,0C11,0030,24A7,0174,9 ;36745       VA_M[TEMP0]+ZLIT0[4]             ; LOAD ADDRESS OF ESP.
                                ;36746
                                ;36747          ;--------------------------------;
                                ;36748          WB_M[MDR].AND.R[LONLIT],         ; CHECK P1LR MBZ BITS.
U 1749, 0C81,2002,0A3D,4047,015A,C ;36749       WX.EQ.0?                         ;
                                ;36750
                                ;36751 =0       ;0------------------------------;
U 15AC, 0C80,0036,4030,0047,00FF,8 ;36752       NEXT/IE.OPER.FAULT               ; RESERVED OPERAND FAULT.
                                ;36753
                                ;36754          ;1------------------------------;
U 15AD, 0D81,2C92,1034,0287,0174,A ;36755       PME_Q_M[MDR].AND.ZLIT24[80]      ; WRITE HARDWARE PME.
                                ;36756
                                ;36757          ;--------------------------------;
                                ;36758          READ.PHY,VA_VA+4,                ; FETCH ESP FROM PCB.
U 174A, 0485,200B,803A,C440,0174,B ;36759       R[P1LR]_M[MDR].ANDNOT.Q          ; LOAD P1LR.
                                ;36760
                                ;36761          ;--------------------------------;
                                ;36762          READ.PHY,VA_VA+4,                ; FETCH SSP FROM PCB.
U 174B, 0C85,2592,4038,4440,0174,C ;36763       R[ESP]_M[MDR]                    ; LOAD ESP.
                                ;36764
                                ;36765          ;--------------------------------;
                                ;36766          READ.PHY,SET MM.NOINT,           ; FETCH USP FROM PCB.
U 174C, 0465,2592,4038,8040,015B,C ;36767       R[SSP]_M[MDR]                    ; LOAD SSP.
                                ;36768
                                ;36769 =0       ;0------------------------------;
                                ;36770          PUSH,                            ; PUSH FOR JUMP TO LS.LDPCTX.SUB1
                                ;36771          R[USP]_M[MDR],                   ;
                                ;36772          PSL<IS.CURM>?,                   ; CHECK IF ON INTERRUPT STACK.
U 15BC, 0085,2592,47B8,C207,0D5C,B 479* ;36773  NEXT/LS.LDPCTX.PSL               ;
                                ;36774
                                ;36775          ;1------------------------------; CATCH RETURN FROM LS.LDPCTX.SUB1
                                ;36776          WRITE M[MDR],SIZE[LONG],         ; PUSH THE NEW PSL ON THE STACK.
U 15BD, 0481,2592,4020,05D8,015C,8 ;36777       NEXT/LS.LDPCTX.PUSH              ;
                                ;36778 =1011
                                ;36779 LS.LDPCTX.PSL:
                                ;36780          ;1011----------------------------; CURRENTLY ON KERNAL STACK.
                                ;36781          VA_M[TEMP0]+ZLIT0[76.],          ; LOAD ADDRESS OF NEW PSL.
U 15CB, 0180,0C11,0032,64A7,0175,3 ;36782       NEXT/LS.LDPCTX.SUB1              ;
                                ;36783
                                ;36784          ;1111----------------------------; CURRENTLY ON INTERRUPT STACK.
U 15CF, 0080,0002,403D,84A7,0174,D ;36785       VA_M[TEMP0]                      ; LOAD ADDRESS OF KSP.
                                ;36786
                                ;36787          ;--------------------------------;
U 174D, 0C80,0036,4030,0040,0174,E ;36788       READ.PHY                         ; FETCH KSP FROM PCB.
                                ;36789
                                ;36790          ;--------------------------------;
U 174E, 0886,15BE,4037,8047,0174,F ;36791       M[TEMP1]_R[SP]                   ; SAVE SP ON ISP.
```

```
                                ;36792          ;---------------------------------;
U 174F, 0084,1592,4039,0047,0175,0      ;36793          R[ISP]_M[TEMP1]                 ;
                                ;36794          ;---------------------------------;
                                ;36795
U 1750, 0886,1036,4030,0087,0175,1      ;36796          M[TEMP1]_PSL                    ;
                                ;36797          ;---------------------------------;
                                ;36798
U 1751, 0180,1C93,8030,2007,0175,2      ;36799          PSL_M[TEMP1].ANDNOT.ZLIT24[4]   ; CLEAR PSL<IS>.
                                ;36800          ;---------------------------------;
                                ;36801
                                ;36802          R[SP]_M[MDR],                   ; LOAD NEW STACK POINTER.
U 1752, 0485,2592,4037,8047,015C,B      ;36803          NEXT/[S.LDPCTX.PSL              ;
                                ;36804
                                ;36805 =0
                                ;36806 LS.LDPCTX.PUSH:
                                ;36807          ;0---------------------------------;
                                ;36808          VA_M[TEMP0]+ZLIT0[72.],         ; LOAD ADDRESS OF NEW PC.
U 15C8, 0180,0C11,0032,44A7,0575,3      ;36809          PUSH,NEXT/LS.LDPCTX.SUB1        ;
                                ;36810          ;1---------------------------------;
                                ;36811          WRITE M[MDR],SIZE[LONG],        ; PUSH THE NEW PC ON THE STACK.
U 15C9, 0081,2592,4120,05D8,003F,9      ;36812          IRD1                            ;
                                ;36813          ;---------------------------------;
                                ;36814
                                ;36815 LS.LDPCTX.SUB1:
                                ;36816          ;---------------------------------;
                                ;36817          READ.PHY,VA_R[SP]_RB-CONX(4),   ; FETCH NEW PSL AND PREPARE TO PUSH.
U 1753, 0084,073C,00A7,84A0,0000,1      ;36818          RETURN [+1]                     ;
```

```
                                ;36819 .TOC "  PROBE, LDPCTX, SVPCTX          : SVPCTX"
                                ;36820
                                ;36821 ;*****************************************************************************
                                ;36822 ;       SVPCTX          Save Process Context
                                ;36823 ;
                                ;36824 ;       Input           PCBB    Physical address of Process Control Block
                                ;36825 ;
                                ;36826 ;       Resources       TEMP0           Hold PCBB on MBUS
                                ;36827 ;                       TEMP1           Temp for PSL and switching stacks.
                                ;36828 ;                       Q               Used to load RNUM for saving GPR'S
                                ;36829 ;                       RNUM            Used to address GPR'S
                                ;36830 ;                       STEPC           Loop count for saving GPR'S
                                ;36831 ;
                                ;36832 ;       Subroutines     LS.MODE.CHECK
                                ;36833 ;
                                ;36834 ;       CHARLIE'S FLOWS 3.7
                                ;36835 ;*****************************************************************************
                                ;36836
                                ;36837 .REGION/IRD1.R1L,IRD1.R1H
                                ;36838 =000
                                ;36839 LS.SVPCTX:
                                ;36840         ;000-----------------------------;
U 03A8, 0D81,AC10,0030,0C87,015C,C ;36841         PC_M[PC]-ZLIT0[1]               ; CORRECT PC FOR NO OPERANDS.
                                ;36842 =
                                ;36843
                                ;36844 .REGION/PRLDSV.R1L,PRLDSV.R1H/PRLDSV.R2L,PRLDSV.R2H/PRLDSV.R3L,PRLDSV.R3H
                                ;36845
                                ;36846 =0       ;0-----------------------------;
                                ;36847         RNUM_Q_ZLIT0[0],                ; PREPARE TO SAVE GPRS.
                                ;36848         PSL<IS.CURM>?,                  ; CHECK FOR PRIVILEGED INSTRUCTION FAULT
U 15CC, 0181,DC37,17B0,0207,0D60,C 479* ;36849         PUSH,NEXT/LS.MODE.CHECK         ;
                                ;36850
                                ;36851         ;1-----------------------------;
                                ;36852         STEPC_14.,                      ; LOAD # OF GPR'S TO SAVE
U 15CD, 0DB0,0C11,0030,84A7,015D,C ;36853         VA_M[TEMP0]+ZLIT0[16.]          ; LOAD PCB ADDRESS OF GPR0
                                ;36854 =0
                                ;36855 LS.SVPCTX.GPRS.2:
                                ;36856         ;0-----------------------------;
                                ;36857         WRITE.PHY R[GPR.R],             ; SAVE GPR(RNUM).
U 15DC, 0480,05BE,403C,C5C8,0175,4 ;36858         NEXT/LS.SVPCTX.GPRS 4           ;
                                ;36859
                                ;36860         ;1-----------------------------;
U 15DD, 0480,05BE,4037,84A7,0175,5 ;36861         VA_R[SP],NEXT/LS.SVPCTX.PC      ; PREPARE TO POP PC OFF OF THE STACK
                                ;36862
                                ;36863 LS.SVPCTX.GPRS.4:
                                ;36864         ;-----------------------------;
                                ;36865         RNUM_Q_Q+1,VA_VA+4,             ; PREPARE TO SAVE NEXT GPR.
                                ;36866         DBZ STEPC?,                     ;
U 1754, 0881,D0B9,133D,8447,015D,C ;36867         NEXT/LS.SVPCTX.GPRS.2           ; LOOP SAVING ALL GPRS.
```

```
                                ;36868 LS.SVPCTX.PC:
                                ;36869          ;-------------------------------;
                                ;36870          READ,SIZE[LONG],                ; POP PC OFF OF THE STACK.
U 1755, 0D80,0C11,0022,44B0,0175,6  ;36871      VA_M[TEMP0]+ZLIT0[72.]          ; LOAD PCB ADDRESS OF SAVED PC.
                                ;36872
                                ;36873          ;-------------------------------;
                                ;36874          SET MM.NOINT,                   ;
U 1756, 0861,2592,4030,05C8,0175,7  ;36875      WRITE.PHY M[MDR]                ; SAVE PC.
                                ;36876
                                ;36877          ;-------------------------------;
U 1757, 0884,073D,0027,84A7,0175,8  ;36878      VA_R[SP]_RB+CONX(4)             ; FINISH POPPING PC.
                                ;36879
                                ;36880          ;-------------------------------;
                                ;36881          READ,SIZE[LONG],                ; POP PSL OFF OF THE STACK.
U 1758, 0580,0C11,0022,64B0,0175,9  ;36882      VA_M[TEMP0]+ZLIT0[76.]          ; LOAD PCB ADDRESS OF SAVED PSL.
                                ;36883
                                ;36884          ;-------------------------------;
U 1759, 0881,2592,4030,05C8,0175,A  ;36885      WRITE.PHY M[MDR]                ; SAVE PSL.
                                ;36886
                                ;36887          ;-------------------------------;
U 175A, 0880,0002,403D,84A7,0175,B  ;36888      VA_M[TEMP0]                     ; LOAD PCB ADDRESS OF KSP.
                                ;36889
                                ;36890          ;-------------------------------;
                                ;36891          R[SP]_RB+CONX(4),               ; FINISH POPPING PSL.
U 175B, 0084,073D,07A7,8207,095E,B 479*  ;36892  PSL<IS.CURM>?                   ; CHECK IF ON INTERRUPT STACK.
                                ;36893
                                ;36894 =1011    ;1011-----------------------------; SWITCH FROM KERNAL TO INTERRUPT STACK
                                ;36895          M[TEMP1]_PSL,                   ; SAVE PSL IN MTEMP.
U 15EB, 0086,1036,4030,0087,0175,C  ;36896      NEXT/LS.SVPCTX.IPL              ;
                                ;36897
                                ;36898          ;1111-----------------------------; ALREADY ON INTERRUPT STACK.
                                ;36899          WRITE.PHY R[KSP],               ; SAVE KSP
U 15EF, 0880,05BE,4038,05C8,0176,0  ;36900      NEXT/LS.SVPCTX.SPS              ;
                                ;36901
                                ;36902 LS.SVPCTX.IPL:
                                ;36903          ;-------------------------------;
                                ;36904          WB_M[TEMP1].AND.ZLIT16[1F],     ; PSL<IPL> <- MAX(1,PSL<IPL>)
U 175C, 0D80,1D12,0A70,F847,015E,8  ;36905      WX.NE.0?                        ;
                                ;36906
                                ;36907 =0       ;0-------------------------------; PSL<IPL>=0.
U 15E8, 0986,1D12,4030,0847,015E,9  ;36908      M[TEMP1]_MB.OR.ZLIT16[1]        ; SET NEW IPL TO 1.
                                ;36909
                                ;36910          ;1-------------------------------; PSL<IPL> .NE. 0.
U 15E9, 0486,05BE,4037,85C8,00A3,7  ;36911      WRITE.PHY R[SP],M[TEMP0]_WB     ; SAVE SP IN PCB AND IN KSP.
                                ;36912
                                ;36913 0A37:    ;*********FORCE ADDRESS*********;
U 0A37, 0884,0592,4038,0047,0175,D  ;36914      R[KSP]_M[TEMP0]                 ; SAVE SP IN KSP
```

```
                                ;36915          ;--------------------------------;
U 175D, 0580,1C92,4030,2007,0175,E  ;36916      PSL_M[TEMP1].OR.ZLIT24[4]        ; SET PSL<IS> = 1.
                                ;36917
                                ;36918          ;--------------------------------;
U 175E, 0086,15BE,4039,0047,0175,F  ;36919      M[TEMP1]_R[ISP]                  ; SWITCH STACKS.
                                ;36920
                                ;36921          ;--------------------------------;
U 175F, 0084,1592,4037,8047,0176,0  ;36922      R[SP]_M[TEMP1]                   ; LOAD NEW STACK POINTER FROM ISP.
                                ;36923
                                ;36924 LS.SVPCTX.SPS:
                                ;36925          ;--------------------------------;
U 1760, 0480,0036,4030,0447,0176,1  ;36926      VA_VA+4                          ; LOAD PCB ADDRESS OF SAVED ESP.
                                ;36927
                                ;36928          ;--------------------------------;
U 1761, 0480,05BE,4038,45C8,0176,2  ;36929      WRITE.PHY R[ESP]                 ; SAVE ESP.
                                ;36930
                                ;36931          ;--------------------------------;
U 1762, 0C80,0036,4030,0447,0176,3  ;36932      VA_VA+4                          ; LOAD PCB ADDRESS OF SAVED SSP.
                                ;36933
                                ;36934          ;--------------------------------;
U 1763, 0480,05BE,4038,85C8,0176,4  ;36935      WRITE.PHY R[SSP]                 ; SAVE SSP.
                                ;36936
                                ;36937          ;--------------------------------;
U 1764, 0C80,0036,4030,0447,0176,5  ;36938      VA_VA+4                          ; LOAD PCB ADDRESS OF SAVED USP.
                                ;36939
                                ;36940          ;--------------------------------;
                                ;36941          WRITE.PHY R[USP],                ; SAVE USP.
U 1765, 0C80,05BE,4138,C5C8,003F,9  ;36942      IRD1                             ;
```

```
                                ;36943 .TOC "  PROBE, LDPCTX, SVPCTX           :       LS.MODE.CHECK"
                                ;36944
                                ;36945 ;*****************************************************************************
                                ;36946 ;       Entry Points    LS.MODE.CHECK
                                ;36947 ;
                                ;36948 ;       Input           PCBB            Saved in MTEMP0
                                ;36949 ;
                                ;36950 ;       Output          TEMP0           PCBB (needed in MTEMP)
                                ;36951 ;
                                ;36952 ;       This is used to check for priviledged instruction faults.
                                ;36953 ;       This routine should be called while branching on PSL<IS.CURM>.
                                ;36954 ;       It will RETURN + 1 if the mode is zero and will take a priviledged
                                ;36955 ;       instruction fault otherwise.
                                ;36956 ;
                                ;36957 ;       CHARLIE'S FLOWS 3.7
                                ;36958 ;*****************************************************************************
                                ;36959
                                ;36960
                                ;36961 =1100
                                ;36962 LS.MODE.CHECK:
                                ;36963         ;1100-------------------------;
                                ;36964         COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG CLEARING TB
                                ;36965         M[TEMP0]_R[PCBB],               ; MOVE PCBB TO AN MTEMP.
U 160C, 0486,05BE,40B9,4057,0000,1  ;36966     RETURN [+1]                     ;
                                ;36967
                                ;36968         ;1101-------------------------;
                                ;36969         CLEAR FLAG3,                    ;
U 160D, 0418,0036,4030,0047,003B,0  ;36970     NEXT/IE.OPCOD.DEC               ; PRIVILEDGED INSTRUCTION FAULT.
                                ;36971
                                ;36972         ;1110-------------------------;
                                ;36973         CLEAR FLAG3,                    ;
U 160E, 0418,0036,4030,0047,003B,0  ;36974     NEXT/IE.OPCOD.DEC               ; PRIVILEDGED INSTRUCTION FAULT.
                                ;36975
                                ;36976         ;1111-------------------------;
                                ;36977         CLEAR FLAG3,                    ;
U 160F, 0418,0036,4030,0047,003B,0  ;36978     NEXT/IE.OPCOD.DEC               ; PRIVILEDGED INSTRUCTION FAULT.
```

```
                                        ;36979 .TOC "  PROBE, LDPCTX, SVPCTX          : Ird Rom Definition"
;36980   .NOBIN
;36981
;36982   .ICODE  ;       OPS             REG                     MEM             OPS         FPA REG                 FPA MEM
;36983        006: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                      ;LDPCTX
;36984             IRD1[NOP][LS.LDPCTX           ]                                [NOP][LS.LDPCTX         ]
;36985   .OCODE
;36986        006: CNTO[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ],
;36987             CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
;36988
;36989
;36990
;36991
;36992   .ICODE
;36993        00C: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                      ;PROBER
;36994             IRD1[LOD][OS.RED              ]                                [LOD][OS.RED            ]
;36995   .OCODE
;36996        00C: CNTO[LOD][OS.RED              ][OS.RED              ]  [LOD][OS.RED              ][OS.RED              ],
;36997             CNT1[NOP][PR.PROBER           ][PR.PROBER           ]  [NOP][PR.PROBER           ][PR.PROBER           ]
;36998
;36999
;37000
;37001
;37002   .ICODE
;37003        00D: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                      ;PROBEW
;37004             IRD1[LOD][OS.RED              ]                                [LOD][OS.RED            ]
;37005   .OCODE
;37006        00D: CNTO[LOD][OS.RED              ][OS.RED              ]  [LOD][OS.RED              ][OS.RED              ],
;37007             CNT1[NOP][PR.PROBEW           ][PR.PROBEW           ]  [NOP][PR.PROBEW           ][PR.PROBEW           ]
;37008
;37009
;37010
;37011
;37012   .ICODE
;37013        007: FPD [NOP][IE.OPCOD.DEC        ]                                [NOP][IE.OPCOD.DEC      ],                      ;SVPCTX
;37014             IRD1[NOP][LS.SVPCTX           ]                                [NOP][LS.SVPCTX         ]
;37015   .OCODE
;37016        007: CNTO[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ],
;37017             CNT1[NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]  [NOP][IE.BAD.IRD          ][IE.BAD.IRD          ]
```

```
;37018  .TOC "  PROBE, LDPCTX, SVPCTX          : Dsize Rom Definition"
;37019  .DCODE
;37020
;37021  006:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;LDPCTX
;37022
;37023  00C:    SIZE [BYTE] [WORD] [BYTE] [   0] [   0] [   0]  ;PROBER
;37024
;37025  00D:    SIZE [BYTE] [WORD] [BYTE] [   0] [   0] [   0]  ;PROBEW
;37026
;37027  007:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;SVPCTX
;37028
;37029  .UCODE
                                        ;37030 .BIN
```

```
                                        ;37031 .TOC "IANDE.MIC"
                                        ;37032 .TOC "REVISION 45.0"
                                        ;37033 ;       Jeff Peng, Gerard Koeckhoven, CHARLIE MCDOWELL
                                        ;37034
;37035  .NOBIN
;37036
;37037  .TOC "  Revision History"
;37038
;37039  ; REV   EXPLAINATION
;37040
;37041  ; 45    Modify machine check(utrap 28:) routine for TB parity error recovery.
;37042  ;
;37043  ; 44    Read modify ACVs go thru read ACV micro trap, and must be
;37044  ;       detected, so just taking a read tbmiss is no good.
;37045  ;       Disable interrupts during logging of machine check.
;37046  ; 43    CANCEL XFC fix of 39
;37047  ;       Fix interval timer problem.
;37048  ;       Make changes for CMODE reserved opcodes (see CMODE revision 20).
;37049  ;       Ensure that a TB-miss and ACC Violation don't cause problems
;37050  ;       Assign permanent addresses to free locations (where possible)
;37051  ; 42    Reassign FEE  with 3EF
;37052  ;       Fix unaligned read across a page boundary to not clobber VA if a fault it taken with FPD=1
;37053  ;       on the second half of the read.
;37054  ;       More on unaligned read bug.  Was causing KSNV because stack flag was left set if tb miss
;37055  ;       routine tried to service an interrupt.
;37056  ;       Fix machine check code to set MM.NOINT before pushing params on stack.
;37057  ; 41    Fix when FPD set & access violation & no TB miss i.e. the PTE is in the TB but access is not
;37058  ;       allowed the MDR will not be clobbered.
;37059  ; 40    Use location 3EE instead of FEE to avoid ROM changes
;37060  ; 39    Fix XFC instruction to Back-up the PC by one and clear all flags before going to WCS.
;37061  ;       Fix "REI" instruction to check for odd PC when entering compatibility mode
;37062  ; 38    5/19/80
;37063  ;       Cleanup and improve the comments.
;37064  ;       5/29/80
;37065  ;       Add special pack routine entry for CVTTP destination length 0.
;37066  ;       6/19/80
;37067  ;       Fix machine check to disable cache on cache parity errors.
;37068  ; 37    Fix FD opcode decode.
;37069  ;       Remove Q_0 from FX.IE.FD.RET.
;37070  ;       Fix WRT BUS ERR double error halt.  Change halt code to 5.
;37071  ;       Clear FPA traps to avoid erroneous trap.
;37072  ;       Set stack flag on halts to tell console to look at front panel.
;37073  ;       Load PC with SCBB+offset on halt due to vector<1:0>=3.
;37074  ;       Fix disabling of cache on cache machine checks.
;37075  ;       Fix console interrupts to properly clear transmitter interrupts.
;37076  ;       Fix handling of FPA detected faults.
;37077  ;       Load VA and set MM.NOINT for interlocked queues with FPD set.
;37078  ;       Fix KSNV to always raise the IPL to 1F.
;37079  ; 36    Initial release.
                                        ;37080         .BIN
```

```
                        ;37081 .TOC "  Interrupts and Exceptions       : INTERRUPT WIDE BRANCH"
                        ;37082
                        ;37083 ;*****************************************************************************
                        ;37084 ;
                        ;37085 ;       THIS IS A TABLE OF THE FIRST INSTRUCTION OF THE INTERRUPT ROUTINES.
                        ;37086 ;       A WIDE BRANCH IS TAKEN TO HERE AND THEN CONTROL IS TRANSFERRED
                        ;37087 ;       TO THE APPROPRIATE ROUTINE.
                        ;37088 ;
                        ;37089 ;       (SEE INDIVIDUAL ROUTINES FOR INPUT AND MODIFY LISTS.)
                        ;37090 ;
                        ;37091 ;*****************************************************************************
                        ;37092
                        ;37093 .SEQUENTIAL
                        ;37094 38:
                        ;37095 IE.INT.WIDE.BRANCH:
                        ;37096 IE.SOFT.INT:
                        ;37097         ;111000--------------------------;
                        ;37098         D_M[SCBB]+ZLIT0[80],              ; LOAD BASE ADDRESS OF SOFT.INT VECTORS.
U 0038, 0D80,EC11,2034,0047,04F4,0      ;37099         PUSH,NEXT/IE.SOFT.IPL             ; WILL RETURN+10(hex) WHICH WRAPS
                        ;37100                                           ; TO LOCATION 08
                        ;37101
                        ;37102 IE.CONSOLE.INT:
                        ;37103         ;111001--------------------------;
                        ;37104         M[TEMP8]_ZLIT16[14],              ; LOAD THE NEW IPL INTO THE NEW PSL.
U 0039, 0186,8D37,0030,A047,00F9,C      ;37105         NEXT/IE.CONSOLE.INT2              ; JUMP TO CONSOLE INTERRUPT ROUTINE.
                        ;37106
                        ;37107 IE.UNIBUS.INT:
                        ;37108         ;111010--------------------------;
                        ;37109         MEMSCAR_ZLIT24[2],                ; LOAD ADDRESS OF UNIBUS INTERRUPT REG.
U 003A, 0D80,0CB7,0030,1687,00F7,7      ;37110         NEXT/IE.UNIBUS.INT2               ; JUMP TO UNIBUS INTERRUPT ROUTINE.
                        ;37111
                        ;37112 IE.INT.TIMER:
                        ;37113         ;111011--------------------------;
                        ;37114         VA_M[SCBB]+ZLIT0[0C0],            ; LOAD THE MACRO VECTOR ADDRESS.
U 003B, 0D50,EC11,0036,04A7,00F6,E      ;37115         SET FLAG2,NEXT/IE.INT.TIMER2      ; NO MORE PARAMETERS TO PUSH.
                        ;37116
                        ;37117 IE.CORRECTED.MEM:
                        ;37118         ;111100--------------------------;
                        ;37119         VA_M[SCBB]+ZLIT0[54],             ; LOAD MACRO VECTOR ADDRESS.
                        ;37120         SET FLAG2,                        ; NO MORE PARAMETERS TO PUSH.
U 003C, 0550,EC11,0032,A4A7,00F4,A      ;37121         NEXT/IE.CORRECTED.MEM2            ;
```

```
                                ;37122 IE.CACHE.PARITY:
                                ;37123          ;111101------------------------; SET ERROR CODE TO PUSH ON THE STACK.
                                ;37124 ;        M[ERRCOD] ZLIT0[3],             ; CHECK IF IN CONSOLE MODE.
U 003D, 0C80,0036,4030,0047,0000,1 ;37125       NEXT/MS.HALT.MICRO              ; NOW ARE EXCEPTIONS*********
                                ;37126
                                ;37127 IE.WRT.BUS.ERR:
                                ;37128          ;111110------------------------;
                                ;37129          VA_M[SCBB]+ZLIT0[60],           ; LOAD MACRO VECTOR ADDRESS.
                                ;37130          SET FLAG2,                      ; ***FLAG CLEARED IN NEXT CYCLE**
U 003E, 0550,EC11,0033,04A7,00F0,0 ;37131       NEXT/IE.WRT.BUS.ERR2            ; JUMP TO WRITE BUS ERROR ROUTINE.
                                ;37132
                                ;37133 IE.POWER.FAIL:
                                ;37134          ;111111------------------------;
                                ;37135          VA_M[SCBB]+ZLIT0[0C],           ; LOAD MACRO VECTOR ADDRESS.
                                ;37136          SET FLAG2,                      ; NO MORE PARAMETERS TO PUSH.
U 003F, 0950,EC11,0030,64A7,00F4,7 ;37137       NEXT/IE.POWER.FAIL2             ; JUMP TO POWER FAIL ROUTINE.
                                ;37138
                                ;37139 .RANDOM
                                ;37140
                                ;37141 8:       ;**HARD ADDR**                  ; CATCH RETURN FROM IE.SOFT.IPL
                                ;37142          ;-----------------------------;
                                ;37143          M[TEMP8] R[TEMP5].RR.16,        ; LOAD IPL INTO PROPER BITS OF NEW PSL.
U 0008, 0C86,8287,0011,4047,00F3,8 ;37144       NEXT/IE.SOFT.INT3               ; CONTINUE WITH SOFTWARE INTERRUPT FLOW.
```

```
;37145 .TOC "  Interrupts and Exceptions       : Initiate Exception or Interrupt"
;37146
;37147 ;*****************************************************************************
;37148 ;       Entry points    IE.ABORT
;37149 ;                       IE.TRAP
;37150 ;                       IE.FAULT
;37151 ;                       IE.INTERRUPT
;37152 ;
;37153 ;       Resources       SP,ISP,KSP,ESP,SSP,USP
;37154 ;                       VA,MDR
;37155 ;                       PSL
;37156 ;                       PC,PCBACK
;37157 ;                       TEMP4           Save old PSL
;37158 ;                       TEMP5           Macro interrupt vector
;37159 ;                       MTEMP8          New PSL
;37160 ;                       MTEMP9          Temp to switch stacks
;37161 ;                       MTEMP10         Temp to switch stacks
;37162 ;                       FLAG0           0 = EXCEPTION
;37163 ;                                       1 = INTERRUPT
;37164 ;                       FLAG1           0 = INTERRUPT MODE
;37165 ;                                       1 = KERNAL MODE
;37166 ;                       FLAG2           0 = MORE PARAMS TO PUSH RETURN TO CALLER
;37167 ;                                       1 = NO MORE PARAMETERS, DO IRD1
;37168 ;                       FLAG3           0 = PUSH PCBACK - 2
;37169 ;                                       1 = PUSH PC
;37170 ;
;37171 ;       THIS IS THE MAIN SUBROUTINE USED TO INITIATE ALL INTERRUPTS AND
;37172 ;       EXCEPTIONS (EXCEPT CHMX).  THE EXECUTION STACK IS SWITCHED TO BE
;37173 ;       EITHER KERNAL OR INTERRUPT, THE OLD PSL AND PC ARE PUSHED ON
;37174 ;       THE NEW STACK AND A NEW PSL AND PC ARE LOADED.
;37175 ;*****************************************************************************
```

```
                                ;37176 .REGION/IANDE.R1L,IANDE.R1H/IANDE.R2L,IANDE.R2H/IANDE.R3L,IANDE.R3H
                                ;37177 0ECC:    ;*********FORCE ADDRESS*********;
                                ;37178 IE.ABORT:
                                ;37179 IE.TRAP:
                                ;37180          ;0------------------------------;
                                ;37181          M[TEMP8]_PSL,                   ; SET IPL IN NEW PST TO CURRENT IPL.
U 0ECC, 0C86,8036,46B0,0087,00EE,8 ;37182       STACK FLAG?,NEXT/IE.EXCEP01     ; CHECK FOR KSNV.
                                ;37183
                                ;37184 0ECD:    ;*********FORCE ADDRESS*********;
                                ;37185 IE.FAULT:
                                ;37186          ;1------------------------------;
                                ;37187          M[TEMP8]_PSL,                   ; SET IPL IN NEW PST TO CURRENT IPL.
                                ;37188          CLEAR TP,                       ; ON FAULTS CLEAR PSL<TP>.
U 0ECD, 0496,8036,46B0,0087,00EE,8 ;37189       STACK FLAG?                     ; CHECK FOR KSNV.
                                ;37190
                                ;37191 =0
                                ;37192 IE.EXCEP01:
                                ;37193          ;0------------------------------;
                                ;37194          M[TEMP8]_MB.AND.0LIT24[4],      ; CLEAR PSL<CM,TP,FPD,CUR> IN NEW PSL.
                                ;37195          CLEAR FLAG0,                    ; CLEAR FLAG TO INDICATE EXCEPTION.
U 0EE8, 0D06,8E92,0030,2047,00ED,6 ;37196       NEXT/IE.EXCEP02                 ;
                                ;37197
                                ;37198          ;1------------------------------;
                                ;37199          M[TEMP9]_R[SP],                 ; BEGIN TO SWITCH STACKS FOR KSNV.
U 0EE9, 0C86,95BE,4037,8047,00F9,8 ;37200       NEXT/IE.KSNV                    ; INVALID KERNAL OR INTERRUPT STACK.
                                ;37201
                                ;37202 IE.EXCEP02:
                                ;37203          ;------------------------------; SET PREV MODE IN NEW PSL TO 11
                                ;37204          M[TEMP8]_MB.OR.ZLIT16[0C0],     ;
U 0ED6, 0D6E,8D12,4036,0047,00ED,A ;37205       SET STACK FLAG                  ;
                                ;37206
                                ;37207          ;------------------------------;
U 0EDA, 0C80,0036,4030,0067,00EE,A ;37208       CLEAR FP TRAPS                  ; AVOID ERRONEOUS TRAP.
                                ;37209
                                ;37210          ;------------------------------;
                                ;37211          READ.PHY,                       ; READ THE MACRO INTERRUPT VECTOR
                                ;37212          CLEAR ARITH TRAPS,              ; AVOID ERRONEOUS TRAP.
U 0EEA, 0C80,0036,4030,00E0,00EE,C ;37213       NEXT/IE.20                      ;
                                ;37214
                                ;37215 IE.INTERRUPT:
                                ;37216          ;------------------------------; SET FLAG TO INDICATE INTERRUPT.
                                ;37217          READ.PHY,SET FLAG0,             ; READ THE MACRO INTERRUPT VECTOR
                                ;37218          CLEAR ARITH TRAPS,              ; AVOID ERRONEOUS TRAP.
U 0EEE, 0840,0036,4680,00E0,00EE,C ;37219       STACK FLAG?                     ; CHECK FOR KSNV.
                                ;37220
                                ;37221          ;*************************************************************************
                                ;37222          ; DETERMINE IF THIS INTERRUPT OR EXCEPTIONIS TO BE HANDLED
                                ;37223          ; ON THE KERNAL STACK, INTERRUPT STACK OR TRANSFER TO WCS.
                                ;37224 =0
                                ;37225 IE.20:   ;0------------------------------;
                                ;37226          R[TEMP5]_M[MDR],                ; SAVE THE MACRO INTERRUPT VECTOR
                                ;37227          CLEAR STACK FLAG,               ;
U 0EEC, 042D,2592,4271,4047,00EF,0 ;37228       WB<1-0>?,NEXT/IE.25             ; BRANCH ON VECTOR<1:0>
```

```
                                ;37229          ;1-----------------------------;
                                ;37230          M[TEMP9] R[SP],                 ; BEGIN TO SWITCH STACKS FOR KSNV.
U 0EED, 0C86,95BE,4037,8047,00F9,B  ;37231      NEXT/IE.RSNV                    ; INVALID KERNAL OR INTERRUPT STACK.
                                ;37232
                                ;37233 =00
                                ;37234 IE.25:   ;00----------------------------; HANDLE ON KS (UNLESS ON IS)
                                ;37235          M[TEMP9] R[SP],                 ; PREPARE TO SWITCH STACKS
                                ;37236          PSL<IS.CURM>?,                  ; BRANCH ON PSL<IS>
U 0EF0, 0C4E,95BE,47B7,8207,08ED,B 479*  ;37237 SET FLAG1,NEXT/IE.KSTACK        ; SET FLAG TO SAY NEW STACK = KS
                                ;37238
                                ;37239          ;01----------------------------; HANDLE ON IS
                                ;37240          M[TEMP9] R[SP],                 ; PREPARE TO SWITCH STACKS
                                ;37241          PSL<IS.CURM>?,                  ; BRANCH ON PSL<IS>
U 0EF1, 040E,95BE,47B7,8207,08EE,B 479*  ;37242 CLEAR FLAG1,NEXT/IE.60          ; CLEAR FLAG TO SAY NEW STACK = IS
                                ;37243
                                ;37244 IE.WCS.BASE:
                                ;37245          ;10----------------------------; HANDLE IN WCS
                                ;37246          M[FPDOFFSET] ZLIT0[8],          ; SET ERROR CODE, AND JUMP TO WCS.
                                ;37247          SET STACK FLAG,                 ; TELL CONSOLE TO LOOK AT FRONT PANEL
U 0EF2, 096E,CC37,0030,4047,0200,1  ;37248      NEXT/2001                       ; GO TO MS.HALT.MICRO (ADR=01) IF NO WCS
                                ;37249
                                ;37250          ;11----------------------------;
                                ;37251          M[FPDOFFSET] ZLIT0[7],          ; SET ERROR CODE
U 0EF3, 0D6E,CC37,0030,3847,00EF,6  ;37252      SET STACK FLAG                  ; TELL CONSOLE TO LOOK AT FRONT PANEL
                                ;37253
                                ;37254          ;-----------------------------;
                                ;37255          PC M[VA]+ZLIT0[2],              ; LOAD SCBB+OFFSET+2 INTO PC
U 0EF6, 0D81,BC11,0030,1487,0000,1  ;37256      NEXT/MS.HALT.MICRO              ; OPERATION UNDEFINED, HALT
                                ;37257
                                ;37258
                                ;37259          ;*****************************************************************************
                                ;37260          ; SWITCH STACKS IF NECESSARY AND SET PREV MODE IN THE NEW PSL.
                                ;37261          ; ALSO SET NEW PSL<IS> IF HANDLING THIS ON THE INTERRUPT STACK.
                                ;37262 =1011
                                ;37263 IE.KSTACK:
                                ;37264          ;1011--------------------------;
                                ;37265          M[TEMP10] R[KSP],               ; FETCH NEW STACK POINTER
                                ;37266          PSL<IS.CURM>?,                  ; BRANCH ON CURMOD
U 0EDB, 0886,A5BE,47B8,0207,08F2,8 479*  ;37267 NEXT/IE.SAVE.SP                  ; TO SAVE OLD STACK POINTER
                                ;37268
                                ;37269 IE.50:   ;1111--------------------------;
                                ;37270          M[TEMP8] MB.OR.ZLIT24[4],       ; SET NEW PSL<IS>=1
U 0EDF, 0586,8C92,4030,2047,00EF,B  ;37271      NEXT/IE.65                      ;
```

```
                                ;37272 =1011
                                ;37273 IE.60:  ;1011-----------------------------;
                                ;37274         M[TEMP8]_MB.OR.ZLIT24[4],       ; SET NEW PSL<IS>=1
U 0EEB, 0586,8C92,4030,2047,00EF,E ;37275      NEXT/IE.ISTACK                  ;
                                ;37276
                                ;37277         ;1111---------------------------;
                                ;37278         M[TEMP8]_MB.OR.ZLIT24[4],       ; SET NEW PSL<IS>=1
U 0EEF, 0586,8C92,4030,2047,00EF,B ;37279      NEXT/IE.65                      ;
                                ;37280
                                ;37281 IE.65:  ;-------------------------------;
                                ;37282         M[TEMP8]_MB.AND.OLIT16[3F],     ; SET NEW PSL<PREV MODE>=0
U 0EFB, 0986,8F12,0031,F847,00ED,7 ;37283      NEXT/IE.PUSH.PSL.PC             ;
                                ;37284
                                ;37285 IE.ISTACK:
                                ;37286         ;-------------------------------;
                                ;37287         M[TEMP10]_R[ISP],               ; FETCH NEW STACK POINTER
                                ;37288         PSL<IS.CURM>?,                  ; BRANCH ON CURMOD
U 0EFE, 0C86,A5BE,47B9,0207,08F2,8 479* ;37289 NEXT/IE.SAVE.SP                 ; TO SAVE OLD STACK POINTER
                                ;37290
                                ;37291 =1*00
                                ;37292 IE.SAVE.SP:
                                ;37293         ;1*00---------------------------;
                                ;37294         R[KSP]_M[TEMP9],                ; SAVE OLD STACK POINTER IN KSP
U 0F28, 0884,9592,4038,0047,00F1,6 ;37295      NEXT/IE.80                      ;
                                ;37296
                                ;37297         ;1*01---------------------------;
                                ;37298         R[ESP]_M[TEMP9],                ; SAVE OLD STACK POINTER IN ESP
U 0F29, 0484,9592,4038,4047,00F0,A ;37299      NEXT/IE.70
                                ;37300
                                ;37301         ;1*10---------------------------;
                                ;37302         R[SSP]_M[TEMP9],                ; SAVE OLD STACK POINTER IN SSP
U 0F2A, 0C84,9592,4038,8047,00F0,E ;37303      NEXT/IE.75
                                ;37304
                                ;37305         ;1*11---------------------------;
                                ;37306         R[USP]_M[TEMP9],                ; SAVE OLD STACK POINTER IN USP
U 0F2B, 0084,9592,4038,C047,00ED,5 ;37307      NEXT/IE.90                      ;
                                ;37308
                                ;37309 IE.70:
                                ;37310         ;-------------------------------;
                                ;37311         M[TEMP8]_MB.AND.OLIT16[7F],     ; SET PREV MODE IN NEW PSL TO EXECUTIVE.
U 0F0A, 0586,8F12,0033,F847,00ED,5 ;37312      NEXT/IE.90                      ;
                                ;37313
                                ;37314 IE.75:
                                ;37315         ;-------------------------------;
                                ;37316         M[TEMP8]_MB.AND.OLIT16[0BF],    ; SET PREV MODE IN NEW PSL TO SUPERVISOR
U 0F0E, 0586,8F12,0035,F847,00ED,5 ;37317      NEXT/IE.90                      ;
                                ;37318
                                ;37319 IE.80:
                                ;37320         ;-------------------------------;
                                ;37321         M[TEMP8]_MB.AND.OLIT16[3F],     ; SET PREV MODE IN NEW PSL TO KERNAL.
U 0F16, 0186,8F12,05F1,F847,00ED,5 ;37322      FLAG1?,NEXT/IE.90               ; IF NEW MODE=KERNAL, DON'T RELOAD SP.
```

```
                        ;37323 =01
                        ;37324 IE.90:
                        ;37325          ;01-------------------------------;
U 0ED5, 0C84,A592,4037,8047,00ED,7      ;37326          R[SP]_M[TEMP10]                 ; SET NEW STACK POINTER.
                        ;37327
                        ;37328 IE.PUSH.PSL.PC:
                        ;37329          ;11-------------------------------;
                        ;37330          VA_R[SP]_RB-CONX(4),            ;
U 0ED7, 046C,073C,0027,84A7,00F1,E      ;37331          SET STACK FLAG                  ; PREPARE TO PUSH ONTO THE STACK.
                        ;37332
                        ;37333          ;-------------------------------;
                        ;37334          M[TEMP4]_PSL,                   ; SAVE OLD PSL TO PUSH ON STACK.
                        ;37335          SET MM.NOINT,                   ; DISABLE MEMORY MANAGEMENT INTERRUPTS
U 0F1E, 0866,4036,46F0,0087,00F0,5      ;37336          WB<31-30>?                      ; CHECK IF WAS IN COMPATABILITY MODE.
                        ;37337
                        ;37338 0F05:    ;*********FORCE ADDRESS*********;
                        ;37339 IE.105:  ;01-------------------------------;
                        ;37340          PSL_M[TEMP8]_MB<31-16>.0,       ; LOAD NEW PSL.
U 0F05, 0886,84B7,0030,0007,00F2,1      ;37341          NEXT/IE.110                     ;
                        ;37342
                        ;37343 0F07:    ;10-------------------------------; PREVIOUS MODE WAS COMPATABILITY MODE.
                        ;37344          PC_M[PCBACK]-CONX(2),           ; MOVE PCBACK-2 TO PC TO CLEAR
                        ;37345          SET FLAG3,                      ; BITS<31-16>, THEN SET FLAG3 TO PUSH PC
U 0F07, 0859,9710,0010,0487,00F0,5      ;37346          NEXT/IE.105                     ; CONTINUE INITIATING THE EXCEP OR INTER
                        ;37347
                        ;37348 IE.110:  ;-------------------------------;
U 0F21, 0B80,067E,F807,F847,00F2,2      ;37349          LONLIT_[3020FF00]               ; LOAD MASK OF PSL MBZ BITS.
                        ;37350
                        ;37351          ;-------------------------------;
                        ;37352          WRITE M[TEMP4].ANDNOT.R[LONLIT],; PUSH OLD PSL ON STACK.
U 0F22, 0C80,4003,84AD,45D8,00ED,8      ;37353          SIZE[LONG].FLAG2?               ; ARE THERE MORE PARAMS TO PUSH?
                        ;37354
                        ;37355 =0*0     ;0*0------------------------------; MORE PARAMS TO PUSH.
                        ;37356          VA_R[SP]_RB-CONX(4),            ; PREPARE TO PUSH ONTO SP.
U 0ED8, 0084,073C,04E7,84A7,04F0,8      ;37357          PUSH,FLAG3?,NEXT/IE.GET.PC      ; IF FLAG3 SET PUSH PC ELSE PCBACK-2
                        ;37358
                        ;37359          ;0*1------------------------------;
                        ;37360          WRITE M[TEMP7],SIZE[LONG],      ; PUSH THE PC OR PCBACK-2 AND RETURN
U 0ED9, 0428,7592,40A0,05D8,0000,1      ;37361          CLEAR STACK FLAG,RETURN [+1]    ; TO PUSH MORE PARAMETERS.
                        ;37362
                        ;37363          ;1*0------------------------------; NO MORE PARAMS TO PUSH.
                        ;37364          VA_R[SP]_RB-CONX(4),            ; PREPARE TO PUSH ONTO SP.
U 0EDC, 0084,073C,04E7,84A7,04F0,8      ;37365          PUSH,FLAG3?,NEXT/IE.GET.PC      ; IF FLAG3 SET PUSH PC ELSE PCBACK-2
                        ;37366
                        ;37367          ;1*1------------------------------;
                        ;37368          WRITE M[TEMP7],SIZE[LONG],      ; PUSH THE PC OR PCBACK-2 AND
U 0EDD, 0C28,7592,4420,05D8,00EF,4      ;37369          CLEAR STACK FLAG,FLAG0?         ; SEE IF NEED TO RAISE IPL TO 1F
                        ;37370
                        ;37371 0EF4:    ;*********FORCE ADDRESS*********;
                        ;37372 IE.LOAD.PC:
                        ;37373          ;0--------------------------------; EXCEPTION
                        ;37374          IPL_[1D],                       ; SET IPL TO CHECK FOR POWER FAIL
U 0EF4, 0580,0D37,05F0,EFA7,00F0,0      ;37375          FLAG1?,NEXT/IE.RAISE.IPL        ;
```

```
                                ;37376 0EF5:   ;*********FORCE ADDRESS*********;
                                ;37377 IE.LOAD.PC01:
                                ;37378         ;1-----------------------------; INTERRUPT OR VECTOR<1:0>=0
                                ;37379         PC_M[TEMP5].ANDNOT.ZLIT0[3],   ; PC _ MACRO VECTOR<31:2>'0<1:0>
U 0EF5, 0980,5C13,8030,1C87,00E4,D ;37380      NEXT/MP.MTPR.EXIT              ;
                                ;37381 =0*
                                ;37382 IE.RAISE.IPL:
                                ;37383         ;0*----------------------------; EXCEPTION AND VECTOR<1:0>=1
U 0F00, 0D86,8D12,4030,F847,00F0,2 ;37384      M[TEMP8]_MB.OR.ZLIT16[1F]      ; RAISE IPL TO 1F.
                                ;37385
                                ;37386 IE.CHECK.DBL.ERR:
                                ;37387         ;1*----------------------------;
U 0F02, 0980,5C13,8030,1C87,00F2,6 ;37388      PC_M[TEMP5].ANDNOT.ZLIT0[3]    ; PC _ MACRO VECTOR<31:2>'0<1:0>
                                ;37389
                                ;37390         ;-----------------------------;
U 0F26, 0C80,0036,4AF0,0047,00F0,1 ;37391      INTPEND OR TIMER?              ; CHECK FOR POWER FAIL OR WRT BUS ERR.
                                ;37392
                                ;37393 =01
                                ;37394 IE.RESTORE.IPL:
                                ;37395         ;01----------------------------; POWER FAIL, LET IT GO THROUGH
                                ;37396         PSL_M[TEMP8],                  ; RESTORE IPL
U 0F01, 0480,8002,403D,8007,00E4,D ;37397      NEXT/MP.MTPR.EXIT              ;
                                ;37398
                                ;37399         ;11----------------------------; NO POWER FAIL,
U 0F03, 0180,0D37,0030,E7A7,00F3,6 ;37400      IPL_[1C]                       ; SET IPL TO CHECK FOR WRT BUS ERROR.
                                ;37401
                                ;37402         ;-----------------------------;
U 0F36, 0480,0036,4030,0047,00F3,A ;37403      NOP                            ; WAIT 1 CYCLE FOR ARBITRATION TO SETTLE
                                ;37404
                                ;37405         ;-----------------------------;
                                ;37406         INTPEND OR TIMER?              ; CHECK FOR WRITE BUS ERR.
U 0F3A, 0C80,0036,4AF0,0047,00F1,5 ;37407
                                ;37408 =01     ;01----------------------------; THERE IS A WRITE BUS ERROR
                                ;37409         MICRO VECTOR?,                 ; BRANCH ON UVCTR SIMPLY TO
U 0F15, 0480,0036,47B0,0047,08EC,F 479* ;37410 NEXT/IE.CLEAR.INT              ; CLEAR THE INTERRUPT
                                ;37411
                                ;37412         ;11----------------------------; THERE IS NOT A WRITE BUS ERROR
                                ;37413         PSL_M[TEMP8],                  ; RESTORE IPL
U 0F17, 0480,8002,403D,8007,00E4,D ;37414      NEXT/MP.MTPR.EXIT              ; AND CONTINUE WITH NORMAL PROCESSING
                                ;37415
                                ;37416 =1111
                                ;37417 IE.CLEAR.INT:
                                ;37418         ;1111--------------------------; THERE IS A WRITE BUS ERROR
U 0ECF, 0080,C592,46F0,0047,00EE,6 ;37419      WB_M[FPDOFFSET],WB<31-30>?     ; CHECK BIT 31 FOR DOUBLE ERROR
                                ;37420
                                ;37421 =10     ;10----------------------------;
                                ;37422         PSL_M[TEMP8],SET_FLAG3,        ; RESTORE PSL, SET FLAG TO PUSH PC
U 0EE6, 0458,8002,403D,8007,00F3,E ;37423      NEXT/IE.WRT.BUS.EXCEP          ;
                                ;37424
                                ;37425         ;11----------------------------;
                                ;37426         M[FPDOFFSET]_ZLIT0[05],        ; ERROR WRITING INTERRUPT STACK, HALT.
                                ;37427         SET_STACK_FLAG,                ; TELL CONSOLE TO LOOK AT FRONT PANEL
U 0EE7, 096E,CC37,0030,2847,0000,1 ;37428      NEXT/MS.HALT.MICRO             ;
```

```
                                        ;37429 IE.WRT.BUS.EXCEP:
                                        ;37430          ;--------------------------------;
                                        ;37431          M[FPDOFFSET]_MB+ZLIT24[40],      ; SET BIT30 TO TRAP DOUBLE WRITE ERRORS
U 0F3E, 0186,CC91,0032,0047,0003,E      ;37432          NEXT/IE.WRT.BUS.ERR              ;
                                        ;37433 =0
                                        ;37434 IE.GET.PC:
                                        ;37435          ;0-------------------------------;
                                        ;37436          RTEMP7_M[PCBACK]-ZLIT0[2],       ; GET PCBACK-2 TO BE PUSHED ON THE STACK
U 0F08, 0585,9C10,00B0,1047,0000,1      ;37437          RETURN~[+1]                      ;
                                        ;37438
                                        ;37439          ;1-------------------------------;
                                        ;37440          RTEMP7_M[PC]-ZLIT0[0],           ; GET PC TO BE PUSHED ON THE STACK.
U 0F09, 0185,AC10,00B0,0047,0000,1      ;37441          RETURN~[+1]                      ;
```

```
                        ;37442 .TOC    "       Interrupts and Exceptions       : SOFTWARE INTERRUPTS"
                        ;37443
                        ;37444 ;*************************************************************************
                        ;37445 ;       Entry points    IE.SOFT.INT3
                        ;37446 ;
                        ;37447 ;       Resources       D               Base address of software interrupts
                        ;37448 ;                       TEMP5           Temporary
                        ;37449 ;                       SISR            Bit causing interrupt is cleared
                        ;37450 ;
                        ;37451 ;       Output          VA              Interrupt vector address
                        ;37452 ;                       MTEMP8          New PSL
                        ;37453 ;                       FLAG2           Set to indicate no more params to push
                        ;37454 ;                       SOFTIPR         IPL of next highest software interrupt
                        ;37455 ;
                        ;37456 ;       Subroutines     IE.SOFT.IPL
                        ;37457 ;                       IE.INTERRUPT
                        ;37458 ;
                        ;37459 ;       The new IPL is calculated from the SISR.  A new PSL is loaded into
                        ;37460 ;       a temp with only the IPL bits set.  The macro vector address is
                        ;37461 ;       4*IPL + SCBB + 80(hex).  The bit is cleared in the SISR and the
                        ;37462 ;       IPL of the next highest software interrupt is loaded into SOFTIPR
                        ;37463 ;       for arbitration by the hardware.
                        ;37464 ;*************************************************************************
                        ;37465
                        ;37466
                        ;37467 ;IE.SOFT.INT:
                        ;37468 ;       ;01000-------------------------;
                        ;37469 ;       D_M[SCBB]+ZLIT0[80],            ; LOAD BASE ADDRESS OF SOFT.INT VECTORS.
                        ;37470 ;       PUSH,NEXT/IE.SOFT.IPL           ;
                        ;37471 ;
                        ;37472 ;       ;11000-------------------------;
                        ;37473 ;       M[TEMP8]_R[TEMP5].RR.16,        ; LOAD IPL INTO PROPER BITS OF NEW PSL.
                        ;37474 ;       NEXT/IE.SOFT.INT3               ;
                        ;37475
                        ;37476 0F38:   ;*********FORCE ADDRESS*********;
                        ;37477 IE.SOFT.INT3:
                        ;37478         ;-------------------------------;
U 0F38, 0484,05F7,0021,4047,00F4,1      ;37479         R[TEMP5]_RB.ASL.2               ; THE VECTOR OFFSET IS IPL*4.
                        ;37480
                        ;37481         ;-------------------------------;
                        ;37482         VA_D+R[TEMP5],                  ; LOAD ADDRESS OF MACRO VECTOR.
U 0F41, 0040,0021,0031,44A7,00EC,E      ;37483         SET FLAG0                       ; THIS IS AN INTERRUPT.
                        ;37484 =0****
                        ;37485         ;0****-------------------------;
                        ;37486         M[SISR]_MB.ANDNOT.ZLITPL[1],    ; CLEAR BIT IN SISR FOR THIS INTERRUPT.
                        ;37487         SET FLAG2,                      ;  NO MORE PARAMETERS TO PUSH.
U 0ECE, 0556,FAD3,8030,0847,04F4,0      ;37488         PUSH,NEXT/IE.SOFT.IPL           ;
                        ;37489
                        ;37490         ;1****-------------------------;
                        ;37491         SOFTIPR_M[TEMP5].RR.16,         ; LOAD SOFTWARE IPR REGISTER USED BY THE
U 0EDE, 0480,53B7,0010,0787,00EE,E      ;37492         NEXT/IE.INTERRUPT               ; HARDWARE FOR ARBITRATION.
                        ;37493 =
```

```
                                ;37494 .TOC "  Interrupts and Exceptions       :       IE.SOFT.IPL"
                                ;37495
                                ;37496 ;*****************************************************************************
                                ;37497 ;       Entry points    IE.SOFT.IPL
                                ;37498 ;
                                ;37499 ;       Resources       PL      Get most significant bit set in SISR
                                ;37500 ;
                                ;37501 ;       Output          TEMP5   IPL of highest software interrupt pending
                                ;37502 ;
                                ;37503 ;       CHARLIE'S FLOWS 2.2
                                ;37504 ;*****************************************************************************
                                ;37505
                                ;37506 0F40:   ;*********FORCE ADDRESS*********;
                                ;37507 IE.SOFT.IPL:
                                ;37508         ;-------------------------------;
U 0F40, 0880,F9C2,403D,8047,00F4,6      ;37509         PL_MSS M[SISR]                  ; LOAD IPL OF HIGHEST SOFTWARE INTERRUPT
                                ;37510
                                ;37511         ;-------------------------------; MOVE IPL INTO A TEMP SO IT CAN BE
                                ;37512         M[TEMP5]_PL,                    ; SHIFTED FOR BUILDING A NEW PSL.
U 0F46, 0C86,5B37,00B0,0047,0001,0      ;37513         RETURN [+16.]                   ;
```

```
                        ;37514 .TOC "  Interrupts and Exceptions       : POWER FAIL"
                        ;37515
                        ;37516 ;*****************************************************************************
                        ;37517 ;       Entry points    IE.POWER.FAIL2
                        ;37518 ;
                        ;37519 ;       Output          TEMP8           New PSL
                        ;37520 ;                       VA              Interrupt vector address
                        ;37521 ;                       FLAG2           Set to indicate no more params to push
                        ;37522 ;
                        ;37523 ;       Load interrupt vector address into VA.  Load new PSL
                        ;37524 ;       into a temp with only IPL bits set.
                        ;37525 ;*****************************************************************************
                        ;37526
                        ;37527 ;IE.POWER.FAIL:
                        ;37528 ;       ;-------------------------------;
                        ;37529 ;       VA_M[SCBB]+ZLIT0[0C],           ; LOAD MACRO VECTOR ADDRESS.
                        ;37530 ;       SET FLAG2,                      ; NO MORE PARAMETERS TO PUSH.
                        ;37531 ;       NEXT/IE.POWER.FAIL2             ; JUMP TO POWER FAIL ROUTINE.
                        ;37532
                        ;37533 0F47:   ;*********FORCE ADDRESS*********;
                        ;37534 IE.POWER.FAIL2:
                        ;37535         ;-------------------------------;
                        ;37536         M[TEMP8]_ZLIT16[1E],            ; SET IPL=1E IN NEW PSL.
U 0F47, 0986,8D37,0030,F047,00EE,E      ;37537         NEXT/IE.INTERRUPT               ; INITIATE THE INTERRUPT.
```

```
                                ;37538 .TOC "  Interrupts and Exceptions       : WRITE BUS ERROR"
                                ;37539
                                ;37540 ;*****************************************************************************
                                ;37541 ;       Entry points    IE.WRT.BUS.ERR2
                                ;37542 ;
                                ;37543 ;       Output          MTEMP8          New PSL
                                ;37544 ;                       VA              Interrupt vector address
                                ;37545 ;                       FLAG2           Set to indicate no more parameters
                                ;37546 ;
                                ;37547 ;       Load interrupt vector address into VA.  Load new PSL
                                ;37548 ;       into a temp with only IPL bits set.
                                ;37549 ;*****************************************************************************
                                ;37550
                                ;37551 ;IE.WRT.BUS.ERR:
                                ;37552 ;       ;--------------------------------;
                                ;37553 ;       VA_M[SCBB]+ZLIT0[60].           ; LOAD MACRO VECTOR ADDRESS.
                                ;37554 ;       SET FLAG2                       ; NO MORE PARAMETERS TO PUSH.
                                ;37555
                                ;37556 =0
                                ;37557 IE.WRT.BUS.ERR2:
                                ;37558         ,0--------------------------------; CLEAR FLAG2 TO REGAIN CONTROL
                                ;37559         M[TEMP8]_ZLIT16[1D],CLEAR FLAG2,; SET IPL=1D IN NEW PSL.
U 0F0C, 0116,8D37,0030,E847,04EE,E ;37560      PUSH,NEXT/IE.INTERRUPT          ; INITIATE THE INTERRUPT.
                                ;37561
                                ;37562         ;1--------------------------------;
U 0F0D, 0D80,0D37,0030,EFA7,00F0,2 ;37563      IPL_[1D],NEXT/IE.CHECK.DBL.ERR  ; CHECK FOR DOUBLE WRITE BUS ERROR
```

```
                        ;37564 .TOC "  Interrupts and Exceptions       : CORRECTED READ DATA ERROR"
                        ;37565
                        ;37566 ;*****************************************************************************
                        ;37567 ;       Entry points    IE.CORRECTED.MEM
                        ;37568 ;
                        ;37569 ;       Output          MTEMP8          New PSL
                        ;37570 ;                       VA              Interrupt vector address
                        ;37571 ;                       FLAG2           Set to indicate no more parameters
                        ;37572 ;
                        ;37573 ;       Load interrupt vector address into VA.  Load new PSL
                        ;37574 ;       into a temp with only IPL bits set.
                        ;37575 ;*****************************************************************************
                        ;37576
                        ;37577 ;IE.CORRECTED.MEM:
                        ;37578 ;       ;-----------------------------------;
                        ;37579 ;       VA_M[SCBB]+ZLIT0[54],               ; LOAD MACRO VECTOR ADDRESS.
                        ;37580 ;       SET FLAG2                           ; NO MORE PARAMETERS TO PUSH.
                        ;37581
                        ;37582 0F4A:   ;*********FORCE ADDRESS*********;
                        ;37583 IE.CORRECTED.MEM2:
                        ;37584         ;-----------------------------------;
                        ;37585         M[TEMP8]_ZLIT16[1A],                ; SET IPL=1A IN NEW PSL.
U 0F4A, 0186,8D37,0030,D047,00EE,E      ;37586         NEXT/IE.INTERRUPT                   ; INITIATE THE INTERRUPT.
```

```
                        ;37587 .TOC "  Interrupts and Exceptions       : MEM ERR, CS PARITY, BAD IRD, TEMP HALT"
                        ;37588 ;*****************************************************************************
                        ;37589 ;       Entry points    IE.MEM.ERROR
                        ;37590 ;                       IE.CS.PARITY
                        ;37591 ;                       IE.BAD.IRD
                        ;37592 ;                       IE.UNUSED.CS
                        ;37593 ;
                        ;37594 ;       Input           FPDOFFSET<31>   Set if in CONSOLE, BOOT or UVERFY code.
                        ;37595 ;                       If FPDOFFSET<31>=1 then FPDOFFSET<2:0> are as follows:
                        ;37596 ;                               FPDOFFSET<2:0>  0 - CONSOLE mode
                        ;37597 ;                               FPDOFFSET<2:0>  1 - UVERFY
                        ;37598 ;                               FPDOFFSET<2:0>  2 - BOOT
                        ;37599 ;                               FPDOFFSET<2:0>  4 - BOOT
                        ;37600 ;                               FPDOFFSET<2:0>  5 - BOOT
                        ;37601 ;
                        ;37602 ;       Output          ERRCOD          Error code
                        ;37603 ;                       RTEMP9          Save MDR to push on the stack
                        ;37604 ;                       VA              Interrupt vector address
                        ;37605 ;                       FLAG2           Cleared to indicate more parameters
                        ;37606 ;
                        ;37607 ;       Resources       SL              Used to extract MSCR's
                        ;37608 ;
                        ;37609 ;       Subroutines     IE.MACH.CHECK01         Save MDR, MA, AND VA
                        ;37610 ;                       IE.ABORT                Initiate the exception
                        ;37611 ;                       IE.MACH.CHECK02         Push machine check params
                        ;37612 ;*****************************************************************************
                        ;37613 ;       TB PARITY ERROR RECOVERY
                        ;37614 ;       If machine gets TB parity error, try to recover from 'SOFT' error by
                        ;37615 ;       forcing TB miss and then retry the fault micro-order.
                        ;37616 ;
                        ;37617 ;       Resources       FPDOFFSET       bit<30> is used as TB P.E. flag
                        ;37618 ;                                       bit<29> is a flag to indicate that
                        ;37619 ;                                       TB P.E. occurs while memory management
                        ;37620 ;                                       is in progress
                        ;37621 ;                       ERRCOD          used as temporary location to save MEMSCR
                        ;37622 ;                       MM.TEMP5        temporary location to save VA
                        ;37623 ;*****************************************************************************
                        ;37624
                        ;37625 28:     ;**UTRAP ADDR**                 ; MEM.ERROR IS DETECTED BY A MICRO TRAP.
                        ;37626 IE.MEM.ERROR:
                        ;37627         ;-------------------------------; SET ERROR CODE TO PUSH ON THE STACK.
                        ;37628         M[ERRCOD] ZLIT0[2],             ;
U 0028, 0D86,BC37,0030,1047,00F5,6      ;37629         NEXT/IE.MEM.ERR10               ; CHECK IF IN CONSOLE MODE.
                        ;37630
                        ;37631 20:     ;**UTRAP ADDR**                 ; CS.PARITY IS DETECTED BY A MICRO TRAP.
                        ;37632 IE.CS.PARITY:
                        ;37633         ;-------------------------------; SET ERROR CODE TO PUSH ON THE STACK.
                        ;37634         M[ERRCOD] ZLIT0[1],             ;
U 0020, 0D86,BC37,0030,0847,00F5,6      ;37635         NEXT/IE.MEM.ERR10               ; CHECK IF IN CONSOLE MODE.
                        ;37636
                        ;37637 3F8:    ;******** FORCE FOR DEFROM DEFAULT***********
                        ;37638 IE.BAD.IRD:
                        ;37639         ;-------------------------------; BRANCH FROM UNUSED IRD ROM SLOT.
                        ;37640         M[ERRCOD] ZLIT0[7],             ; SET ERROR CODE TO PUSH ON THE STACK.
U 03F8, 0186,BC37,0030,3847,00F5,6      ;37641         NEXT/IE.MEM.ERR10               ; CHECK IF IN CONSOLE MODE.
```

```
                                        ;37642
                                        ;37643 3F9:     ;******** FORCE FOR IRD1 DEFAULT***********
                                        ;37644 IE.IRD1.ERROR:
                                        ;37645          ;--------------------------------; AN IRD1 FAILED TO USE THE IRD1 ROM.
                                        ;37646          M[ERRCOD]_ZLIT0[6],              ; SET ERROR CODE TO PUSH ON THE STACK.
U 03F9, 0586,BC37,0030,3047,00F5,6      ;37647          NEXT/IE.MEM.ERR10                ; CHECK IF IN CONSOLE MODE.
```

```
                                ;37648 17F9:    ;*********FORCE ADDRESS*********;
                                ;37649 IE.UNUSED.CS:
                                ;37650          ;-------------------------------;
                                ;37651          R[R6]_M[ERRCOD],                ; MOVE ERROR CODE INTO GPR FOR ACCESS
U 17F9, 0484,B592.4035,8047,00F5,6 ;37652       NEXT/IE.MEM.ERR10               ; FROM THE CONSOLE.
                                ;37653
                                ;37654 0F56:    ;*********FORCE ADDRESS*********;
                                ;37655 IE.MEM.ERR10:
                                ;37656          ;-------------------------------;
                                ;37657          WB_M[FPDOFFSET],                ; CHECK IF IN CONSOLE MODE
U 0F56, 0080,C592.46F0,0047,00F7,D ;37658       WB<31-30>?                      ;
                                ;37659 =00
                                ;37660 =01      ;01-----------------------------; NOT IN CONSOLE MODE
                                ;37661          MEMSCAR_ZLIT24[8],              ; LOAD ADDRESS OF MCESR, DISABLE INTS
                                ;37662          PATCH,                          ; ** jump to PCS to check if TB parity error
U 0F7D, 1D80,0CB7,0030,4687,00F4,8 ;37663       NEXT/IE.CHECK.FOR.CACHE         ;
                                ;37664
                                ;37665 =11      ;11-----------------------------;
U 0F7F, 0480,C592,4230,0047,00EF,8 ;37666       WB_M[FPDOFFSET],WB<5-0>?        ; RETURN TO CONSOLE, BOOT, OR UVERIFY.
                                ;37667
                                ;37668 0EF8:    ;*********FORCE ADDRESS*********;
                                ;37669 IE.RET.TO.CN.BT.MV:
                                ;37670          ;111000-------------------------; RETURN TO CONSOLE.
                                ;37671          M[TEMP7]_ZLIT24[20],STEPC_2,    ; BAD READ/WRITE TO MEMORY
U 0EF8, 01A6,7CB7,0031,0047,0096,C ;37672       NEXT/CN.ERROR                   ;
                                ;37673
                                ;37674 0EF9:    ;111001-------------------------;
U 0EF9, 0880,0036,40B0,0047,0000,1 ;37675       RETURN [+1]                     ; RETURN TO MICRO VERIFY
                                ;37676
                                ;37677 0EFA:    ;111010-------------------------; RETURN TO BOOT.
                                ;37678          VA_R[TEMP0]_M[VA]-CONX(4),      ; POINT BACK TO OFFENDING WORD
U 0EFA, 0C85,B710.00A0,04A7,0000,2 ;37679       RETURN [+2]                     ;
                                ;37680
                                ;37681 0EFC:    ;111100-------------------------; RETURN TO BOOT.
U 0EFC, 0880,0036,40B0,0047,0000,2 ;37682       RETURN [+2]                     ;
                                ;37683
                                ;37684 0EFD:    ;111101-------------------------; RETURN TO BOOT.
                                ;37685          VA_Q_D_D+ZLIT8[2],              ; POINT TO NEXT PAGE
U 0EFD, 0580,0DB1,30B0,14A7,0000,1 ;37686       RETURN [+1]                     ;
```

```
                                ;37687 0F4B:
                                ;37688 IE.CHECK.FOR.CACHE:
                                ;37689          ;################################; ** code replaced in PCs **
U 0F4B, 0486,9036,4030,0647,00F5,9 ;37690       M[TEMP9]_MEMSCR                 ; READ MCESR TO CHECK FOR CACHE ERROR.
                                ;37691
                                ;37692          ;################################;
U 0F59, 0180,9C92,0A30,4047,0AB4,B 335* ;37694  WB_M[TEMP9].AND.ZLIT24[08],      ; CHECK BUS ERROR BIT WHICH
                                ;37695          WX.EQ.0?                        ; INCLUDES CACHE PARITY ERRORS
                                ;37696 2B4B:    ;==============================; ** code in PCS **
U 2B4B, 0D80,BC10,0A30,1047,0AB4,C 399* ;37698  WB_M[ERRCOD]-ZLIT0[2],          ; CHECK IF IT'S A BUS OR TB MACHINE
                                ;37699          WX.EQ.0?                        ; CHECK
                                ;37700 2B4C:    ;0=============================;
U 2B4C, 0480,0036,4030,0047,02B6,C ;37701       NEXT/PCS.IE.CHECK.FOR.CACHE     ; NOT A BUS OR TB ERROR
                                ;37702
                                ;37703 2B4D:    ;1=============================;
                                ;37704          WB_M[FPDOFFSET],                ; CHECK FOR TB PARITY ERROR FLAG,
U 2B4D, 0880,C592,46F0,0047,0AB4,E 330* ;37705  WB<31-30>?                       ; BIT FPDOFFSET<30>
                                ;37706
                                ;37707 2B4E:    ;10============================; TB FLAG ISN'T SET
                                ;37708          M[ERRCOD]_MEMSCR,               ; READ MEMSCR TO CHECK FOR TB ERROR
U 2B4E, 0086,B036,4030,0647,02B5,0 ;37709       NEXT/PCS.IE.TB.PE               ; ERRCOD IS A TEMPORARY LOCATION
                                ;37710
                                ;37711 2B4F:    ;11============================;
U 2B4F, 0480,0036,4030,0047,02B6,C ;37712       NEXT/PCS.IE.CHECK.FOR.CACHE     ; TB FLAG IS SET, RETURN TO NORMAL
                                ;37713                                          ; MACHINE CHECK FLOW
                                ;37714
                                ;37715 2B50:
                                ;37716 PCS.IE.TB.PE:
                                ;37717          ;==============================;
                                ;37718          WB_M[ERRCOD].AND.ZLIT24[0A],    ; CHECK FOR ERRORS OTHER THAN TB P.E.
U 2B50, 0D80,BC92,0A70,5047,0AB5,2 329* ;37719  WX.NE.0?
                                ;37720
                                ;37721 2B52:    ;0=============================;
                                ;37722          M[FPDOFFSET]_MB.OR.ZLIT28[04],  ; IT IS THE FIRST TB PARITY ERROR
U 2B52, 0986,CC52,4030,2047,02B5,1 ;37723       NEXT/PCS.IE.TB.PE.1             ; SET TB FLAG
                                ;37724
                                ;37725 2B53:    ;1=============================;
                                ;37726          M[ERRCOD]_ZLIT0[2],             ; OTHER THAN TB PARITY ERRCOD
U 2B53, 0D86,BC37,0030,1047,02B6,C ;37727       NEXT/PCS.IE.CHECK.FOR.CACHE     ; SET ERRCOD TO INDICATE BUS/U.B.
                                ;37728                                          ; UNAL/CACHE ERROR, RETURN TO NORMAL
                                ;37729                                          ; MACHINE ROUTINE
```

```
                                        ;37730 2B51:
                                        ;37731 PCS.IE.TB.PE.1:
                                        ;37732          ;================================;
U 2B51, 0480,05BE,403D,8607,02B5,4      ;37733          MEMSCR_R[ZERO]                  ; CLEAN UP MCESR
                                        ;37734
                                        ;37735 2B54:    ;================================;
                                        ;37736          WB_M[FPDOFFSET].AND.ZLIT24[20],  ; CHECK FOR MM PTE BUS CYCLE FLAG
U 2B54, 0180,CC92,0A31,0047,0AB5,6 335* ;37737          WX.EQ.0?
                                        ;37738
                                        ;37739 2B56:    ;0===============================;
U 2B56, 0C80,0036,4030,0047,02B6,1      ;37740          NEXT/PCS.IE.TB.PTE              ; MUST BE MM PxPTE BUS CYCLE
                                        ;37741
                                        ;37742 2B57:    ;1===============================;
                                        ;37743          WB_M[ERRCOD].AND.ZLIT24[01],    ; MUST BE I OR D STREAM BUS CYCLE
U 2B57, 0D80,BC92,0A30,0847,0AB5,8 335* ;37744          WX.EQ.0?                        ; CHECK FOR XB FETCH
                                        ;37745
                                        ;37746 2B58:    ;0===============================;
                                        ;37747          R[MM.TEMP5]_M[VA],              ; IS XB FETCH, SAVE VA
U 2B58, 0885,B592,4033,8047,02B5,B      ;37748          NEXT/PCS.IE.TB.PE.2
                                        ;37749
                                        ;37750 2B59:    ;1===============================;
                                        ;37751          PUSH,                           ; CALL SUBROUTINE TO INVALIDATE TB
U 2B59, 0C80,0036,4030,0047,06B6,3      ;37752          NEXT/PCS.IE.TB.PE.INVALIDATE
                                        ;37753
                                        ;37754 2B55:    ;================================;
U 2B55, 0C81,A592,4030,0487,02B5,A      ;37755          FLUSH XB
                                        ;37756
                                        ;37757 2B5A:    ;================================;
U 2B5A, 0080,0036,40B0,0047,0000,0      ;37758          RETURN [+0]                     ; RETRY FAULT UCODE
                                        ;37759
                                        ;37760 2B5B:
                                        ;37761 PCS.IE.TB.PE.2:
                                        ;37762          ;================================;
                                        ;37763          VA_M[PC]+R[ZERO],               ; SET UP NEW VA
                                        ;37764          PUSH,                           ; GO TO INVALIDATE TB
U 2B5B, 0881,A001,003D,84A7,06B6,3      ;37765          NEXT/PCS.IE.TB.PE.INVALIDATE
                                        ;37766
                                        ;37767 2B5C:    ;================================;
                                        ;37768          VA_M[PC]+ZLIT0[04],             ; SET UP VA
                                        ;37769          PUSH,                           ; GO TO INVALIDATE TB
U 2B5C, 0D81,AC11,0030,24A7,06B6,3      ;37770          NEXT/PCS.IE.TB.PE.INVALIDATE
                                        ;37771
                                        ;37772 2B5D:    ;================================;
U 2B5D, 0880,05BE,4033,84A7,02B5,E      ;37773          VA_R[MM.TEMP5]                  ; RESTORE ORIGINAL VA
                                        ;37774
                                        ;37775 2B5E:    ;================================;
U 2B5E, 0C81,A592,4030,0487,02B5,F      ;37776          FLUSH XB
                                        ;37777
                                        ;37778 2B5F:    ;================================;
U 2B5F, 0480,0036,4030,0047,02B6,0      ;37779          NOP                             ; WAIT FOR FLUSHING COMPLETE
                                        ;37780
                                        ;37781 2B60:    ;================================;
                                        ;37782          RETURN AND INHIBIT DESTINATIONS,; ALL DESTINATIONS INHIBITED
U 2B60, 0480,0036,40F0,0047,0000,0      ;37783          NEXT/0                          ; RETRY THE FAULT UCODE
```

```
                                ;37784 2B61:
                                ;37785 PCS.IE.TB.PTE:
                                ;37786          ;================================;
                                ;37787          M[FPDOFFSET]_MB.ANDNOT.ZLIT28[2],; CLEAR MM PxPTE BUS CYCLE FLAG
                                ;37788          PUSH,                           ; GO FLUSH XB FOR POSSIBLE NONHIT
U 2B61, 0186,CC53,8030,1047,06B6,3 ;37789       NEXT/PCS.IE.TB.PE.INVALIDATE    ; TAG GROUP MACHINE CHECK
                                ;37790
                                ;37791 2B62:    ;================================;
U 2B62, 0080,0036,40B0,0047,03FF,F ;37792       RETURN [-1]                     ; SPECIAL MM PxPTE BUS CYCLE RETURN
                                ;37793                                          ; TO MM.GET.PTE20 OR
                                ;37794                                          ; TO MM.GET.WRITE.PTE32
                                ;37795 2B63:
                                ;37796 PCS.IE.TB.PE.INVALIDATE:
                                ;37797          ;================================;
U 2B63, 0480,0036,4030,0057,02B6,4 ;37798       COMPLETE CPU BUS CYCLES
                                ;37799
                                ;37800 2B64:    ;================================;
U 2B64, 0980,0CB7,0030,1E87,02B6,5 ;37801       MEMSCAR_ZLIT24[3]               ; LOAD ADDRESS OF TB GDR
                                ;37802
                                ;37803 2B65:    ;================================;
U 2B65, 0086,B036,4030,0647,02B6,6 ;37804       M[ERRCOD]_MEMSCR                ; SAVE TB GDR
                                ;37805
                                ;37806 2B66:    ;================================;
U 2B66, 0980,0CB7,0030,5607,02B6,7 ;37807       MEMSCR_ZLIT24[0A]               ; STEER GROUP 0
                                ;37808
                                ;37809 2B67:    ;================================;
U 2B67, 0481,B5BE,403D,8507,02B6,8 ;37810       TB_R[ZERO]                      ; INVALIDATE TB GPR0
                                ;37811
                                ;37812 2B68:    ;================================;
U 2B68, 0980,0CB7,0030,6E07,02B6,9 ;37813       MEMSCR_ZLIT24[0D]               ; STEER GROUP 1
                                ;37814
                                ;37815 2B69:    ;================================;
U 2B69, 0C81,B5BE,403D,8507,02B6,A ;37816       TB_R[ZERO]                      ; INVALIDATE TB GPR1
                                ;37817
                                ;37818 2B6A:    ;================================;
U 2B6A, 0480,B592,4030,0607,02B6,B ;37819       MEMSCR_M[ERRCOD]                ; RESTORE TB GDR
                                ;37820
                                ;37821 2B6B:    ;================================;
U 2B6B, 0880,0036,40B0,0047,0000,1 ;37822       RETURN [+1]
                                ;37823
```

```
                                ;37824 2B6C:
                                ;37825 PCS.IE.CHECK.FOR.CACHE:
                                ;37826          ;================================;
U 2B6C, 0C86,9036,4030,0647,02B6,D ;37827       M[TEMP9]_MEMSCR                  ; READ MCESR TO CHECK FOR CACHE ERROR
                                ;37828
                                ;37829 2B6D:    ;================================; RETURN CONTROL TO CCS
                                ;37830          WB_M[TEMP9].AND.ZLIT24[08],      ; CHECK BUS ERRROR BIT WITH
U 2B6D, 0580,9C92,0A70,4047,00BF,A ;37831       WX.NE.0?                         ; INCLUDE CACHE PARITY ERROR
                                ;37832 IE.CHECK.FOR.CACHE.CONTINUE:
                                ;37833 0BFA:    ;0********FORCE ADDRESS*********; NOT BUS ERROR (IE NOT CACHE ERROR)
                                ;37834          SET MM.NOINT,                    ; DISABLE INTERRUPTS
U 0BFA, 0C60,0036,43F0,0047,00ED,0 ;37835       FPD?,NEXT/IE.MACH.CHK.RBACK      ;
                                ;37836
                                ;37837 0BFB:    ;1********FORCE ADDRESS*********; BUS ERROR, CHECK FOR CACHE ERROR
                                ;37838          SET MM.NOINT,                    ; DISABLE INTERRUPTS
U 0BFB, 0960,0CB7,0030,2687,00BF,7 ;37839       MEMSCAR_ZLIT24[4]                ; LOAD ADDRESS OF CACHE ERROR REGISTER
```

```
                                ;37840 0BF7:    ;*********FORCE ADDRESS*********;
U 0BF7, 0C86,9036,4030,0647,00BF,C      ;37841          M[TEMP9]_MEMSCR                 ; READ CACHE ERROR REGISTER
                                ;37842
                                ;37843 0BFC:    ;*********FORCE ADDRESS*********;
                                ;37844          WB_M[TEMP9].AND.ZLIT24[0C],     ; CHECK IF TAG OR DATA CACHE ERROR
U 0BFC, 0580,9C92,0A70,6047,00F2,4      ;37845          WX.NE.0?                        ;
                                ;37846
                                ;37847 =0
                                ;37848 IE.MEM.ERR20:
                                ;37849          ;0------------------------------; NOT CACHE ERROR
U 0F24, 0480,0036,43F0,0047,00ED,0      ;37850          FPD?,NEXT/IE.MACH.CHK.RBACK     ; ATTEMPT TO PACK OR ROLL BACK.
                                ;37851
                                ;37852          ;1------------------------------; CACHE ERROR DISABLE THE CACHE
U 0F25, 0580,0CB7,0030,3687,00F5,C      ;37853          MEMSCAR_ZLIT24[6]               ; LOAD ADDRESS OF CACHE DISABLE REGISTER
                                ;37854
                                ;37855          ;------------------------------;
                                ;37856          MEMSCR_ZLIT24[1],               ; DISABLE THE CACHE.
U 0F5C, 0580,0CB7,0030,0E07,00F2,4      ;37857          NEXT/IE.MEM.ERR20               ;
                                ;37858
                                ;37859 =000
                                ;37860 IE.MACH.CHK.RBACK:
                                ;37861          ;000-----------------------------;
                                ;37862          R[TEMP0]_RBSP,                  ; SAVE RBSP TO LOAD INTO THE STEPC.
                                ;37863          RBSP.EQ.0?,                     ; CHECK FOR RBSP = 0.
U 0ED0, 0885,E036,4BB0,0047,04F5,A      ;37864          PUSH,NEXT/IE.RBS.RBACK          ; ROLL BACK THE REGISTER SIDE EFFECTS.
                                ;37865
                                ;37866          ;001-----------------------------;
                                ;37867          M[FPDOFFSET]_MB+ZLIT8[5],       ; SET CODE FOR RETURN FROM FPD.PACK.
U 0ED1, 0986,CD91,0230,2847,08FC,0 377* ;37868          WB<5-0>?,NEXT/IE.FPD.PACK       ; FPD WAS SET, PACK IT UP.
                                ;37869
                                ;37870 IE.MACH.CHK.FAULT:
                                ;37871          ;010-----------------------------; CLEAR FLAG TO PUSH PCBACK-2
                                ;37872          R[TEMP9]_M[MDR],CLEAR FLAG3,    ; SAVE MDR, PC, AND VA TO
U 0ED2, 0C1D,2592,4032,4047,04F5,E      ;37873          PUSH,NEXT/IE.MACH.CHECK01       ; PUSH ON THE STACK.
                                ;37874
                                ;37875          ;101-----------------------------;
                                ;37876          VA_M[SCBB]+ZLIT0[4],            ; LOAD MACRO VECTOR ADDRESS.
                                ;37877          CLEAR FLAG2,                    ; MORE PARAMETERS TO PUSH.
U 0ED3, 0D10,EC11,0030,24A7,04EC,D      ;37878          PUSH,NEXT/IE.FAULT              ; INITIATE THE EXCEPTION.
                                ;37879
                                ;37880          ;110-----------------------------; LOAD SIZE OF MSCR'S(SEE IE.WRITE.MSCR)
                                ;37881          SL_[4],                         ; PUSH MACHINE CHECK PARAMETERS,
U 0ED4, 0980,0F76,4030,2047,00F7,6      ;37882          NEXT/IE.MACH.CHECK02            ; AND BEGIN THE MACRO EXCEPTION HANDLER.
                                ;37883 =
                                ;37884 0002:
                                ;37885 GL.NOP.IRD1:
                                ;37886          ;------------------------------;
U 0002, 0080,0036,4130,0047,003F,9      ;37887          IRD1                            ;
                                ;37888
                                ;37889 17FF:
                                ;37890 GL.DBG.NOP:
                                ;37891          ;------------------------------;
U 17FF, 0480,0036,4030,0047,017F,F      ;37892          NEXT/GL.DBG.NOP                 ;
```

```
                                ;37893 .TOC "  Interrupts and Exceptions       : MACHINE CHECK ROUTINES"
                                ;37894 .TOC "  Interrupts and Exceptions       :       IE.MACH.CHECK01"
                                ;37895
                                ;37896 ;*************************************************************************
                                ;37897 ;       Entry points    IE.MACH.CHECK01
                                ;37898 ;
                                ;37899 ;       Output          RTEMP10         Save PC
                                ;37900 ;                       RTEMP11         Save VA
                                ;37901 ;                       RTEMP12         Save Saved Mode Register
                                ;37902 ;
                                ;37903 ;       Save MDR, MA, and VA to push on the stack.
                                ;37904 ;*************************************************************************
                                ;37905
                                ;37906 IE.MACH.CHECK01:
                                ;37907         ;-------------------------------;
U 0F5E, 0085,A592,4032,8047,00F6,0  ;37908         R[TEMP10]_M[PC]                 ;
                                ;37909
                                ;37910         ;-------------------------------;
U 0F60, 0885,B592,4032,C047,00F6,2  ;37911         R[TEMP11]_M[VA]                 ;
                                ;37912
                                ;37913         ;-------------------------------;
U 0F62, 0D80,0CB7,0030,0E87,00F6,C  ;37914         MEMSCAR_ZLIT24[1]               ; LOAD ADDRESS OF SAVED MODE REGISTER
                                ;37915
                                ;37916         ;-------------------------------;
                                ;37917         R[TEMP12]_MEMSCR,               ;
U 0F6C, 0C84,0036,40B3,0647,0000,1  ;37918         RETURN [+1]                     ;
```

```
                                ;37919 .TOC "  Interrupts and Exceptions       :       IE.MACH.CHECK02"
                                ;37920
                                ;37921 ;*****************************************************************************
                                ;37922 ;       Entry points    IE.MACH.CHECK02
                                ;37923 ;
                                ;37924 ;       Input           RTEMP8          Saved WDR
                                ;37925 ;                       RTEMP9          Saved MDR
                                ;37926 ;                       RTEMP10         Saved MA
                                ;37927 ;                       RTEMP11         Saved VA
                                ;37928 ;                       RTEMP12         Saved "Saved Mode Register"
                                ;37929 ;
                                ;37930 ;       Resources       MEMSCAR
                                ;37931 ;                       VA
                                ;37932 ;                       STACKFLAG
                                ;37933 ;
                                ;37934 ;       Subroutines     IE.WRITE.MSCR   Push MSCR(MSCAR) on the stack
                                ;37935 ;                       IE.LOAD.PC+1    Finish initiating the mach check
                                ;37936 ;                       MS.DEC.SP       Decrement the SP
                                ;37937 ;
                                ;37938 ;       This routine pushes the machine check parameters onto the stack.
                                ;37939 ;       The parameters pushed are:
                                ;37940 ;
                                ;37941 ;               Length Parameter
                                ;37942 ;               Error Code
                                ;37943 ;               VA
                                ;37944 ;               PC at time of error
                                ;37945 ;               MDR
                                ;37946 ;               Saved Mode Register (MSCR #1)
                                ;37947 ;               READ Lock Timeout Register (MSCR #2)
                                ;37948 ;               TB Group Parity Error Register (MSCR #D)
                                ;37949 ;               Cache Error Register (MSCR #4)
                                ;37950 ;               BUS Error Register (MSCR #9)
                                ;37951 ;               Machine Check Error Summary Register (MSCR #8)
                                ;37952 ;               PC
                                ;37953 ;               PSL
                                ;37954 ;*****************************************************************************
                                ;37955
                                ;37956 IE.MACH.CHECK02:
                                ;37957          ;-------------------------------;
U 01 76, 0980,UEF6,4030,C047,00EE,0     ;37958          PL_[24.]                        ; LOAD POSITION OF MSCR'S ON WBUS.
                                ;37959
                                ;37960 =000     ;000----------------------------;
                                ;37961          MEMSCAR_ZLIT24[8],              ; PUSH THE MACHINE CHECK ERROR SUMMARY
U 0EE0, 0580,0CB7,0030,4687,04F3,0      ;37962          PUSH,NEXT/IE.WRITE.MSCR         ; REGISTER ON THE STACK
```

```
                                ;37963          ;001----------------------------;
                                ;37964          MEMSCAR_ZLIT24[9],              ; LOAD ADDRESS OF BUS ERROR REGISTER
U 0EE1, 0180,0CB7,0030,4E87,04F3,0  ;37965      PUSH,NEXT/IE.WRITE.MSCR         ; PUSH BUS ERROR REGISTER ON THE STACK
                                ;37966
                                ;37967          ;010----------------------------;
                                ;37968          MEMSCAR_ZLIT24[4],              ; LOAD ADDRESS OF CACHE ERROR REGISTER
U 0EE2, 0980,0CB7,0030,2687,04F3,0  ;37969      PUSH,NEXT/IE.WRITE.MSCR         ; PUSH CACHE ERROR REGISTER ON THE STACK
                                ;37970
                                ;37971          ;011----------------------------; PUSH TB GROUP PARITY ERROR REGISTER
                                ;37972          MEMSCAR_ZLIT24[0D],             ; TB GROUP PARITY ERROR REGISTER
U 0EE3, 0980,0CB7,0030,6E87,04F3,0  ;37973      PUSH,NEXT/IE.WRITE.MSCR         ; ON THE STACK
                                ;37974
                                ;37975          ;100----------------------------;
                                ;37976          MEMSCAR_ZLIT24[02],             ; LOAD ADDRESS OF READ LOCK TIMEOUT.
U 0EE4, 0580,0CB7,0030,1687,04F3,0  ;37977      PUSH,NEXT/IE.WRITE.MSCR         ; PUSH READ LOCK TIMEOUT ON THE STACK.
                                ;37978
                                ;37979          ;101----------------------------;
                                ;37980          VA_R[SP]_RB-CONX(4),            ;
U 0EE5, 0C6C,073C,0027,84A7,00F2,0  ;37981      SET STACK FLAG                  ; PREPARE TO PUSH ONTO THE STACK
                                ;37982 =
                                ;37983 =0       ;0-----------------------------;
                                ;37984          M[TEMP9]_R[TEMP12],             ; PUSH SAVED MODE REGISTER ON THE STACK
                                ;37985          SET STACK FLAG,                 ;
U 0F2C, 046E,95BE,4033,0047,04F3,1  ;37986      PUSH,NEXT/IE.WRITE.MSCR.0       ;
                                ;37987
                                ;37988          ;1------------------------------;
                                ;37989          VA_R[SP]_RB-CONX(4),            ;
U 0F2D, 0C6C,073C,0027,84A7,00F1,0  ;37990      SET STACK FLAG                  ; PREPARE TO PUSH ONTO THE STACK
                                ;37991
                                ;37992 =000     ;000----------------------------;
                                ;37993          WRITE R[TEMP9],SIZE[LONG],      ; PUSH THE SAVED MDR ON THE STACK
U 0F10, 0080,05BE,4022,45D8,04CB,3  ;37994      PUSH,NEXT/MS.DEC.SP             ; PREPARE TO PUSH ONTO THE STACK
                                ;37995
                                ;37996          ;001----------------------------;
                                ;37997          WRITE R[TEMP10],SIZE[LONG],     ; PUSH THE SAVED PC ON THE STACK
U 0F11, 0080,05BE,4022,85D8,04CB,3  ;37998      PUSH,NEXT/MS.DEC.SP             ; PREPARE TO PUSH ONTO THE STACK
                                ;37999
                                ;38000          ;010----------------------------;
                                ;38001          WRITE R[TEMP11],SIZE[LONG],     ; PUSH THE SAVED VA ON THE STACK
U 0F12, 0480,05BE,4022,C5D8,04CB,3  ;38002      PUSH,NEXT/MS.DEC.SP             ; PREPARE TO PUSH ONTO THE STACK
                                ;38003
                                ;38004          ;011----------------------------;
                                ;38005          WRITE M[ERRCOD],SIZE[LONG],     ; PUSH THE ERROR CODE ON THE STACK
U 0F13, 0080,B592,4020,05D8,04CB,3  ;38006      PUSH,NEXT/MS.DEC.SP             ; PREPARE TO PUSH ON THE STACK
                                ;38007
                                ;38008          ;100----------------------------;
                                ;38009          WRITE ZLIT0[28],SIZE[LONG],     ; PUSH THE LENGTH PARAMETER ON THE STACK
                                ;38010          CLEAR STACK FLAG,               ;
U 0F14, 0528,0C37,0421,45D8,00EF,4  ;38011      FLAG0?,NEXT/IE.LOAD.PC          ; GO LOAD NEW PC AND PSL.
                                ;38012 =
```

```
                        ;38013 .TOC "  Interrupts and Exceptions       :       IE.WRITE.MSCR"
                        ;38014
                        ;38015 ;*****************************************************************************
                        ;38016 ;       Entry points    IE.WRITE.MSCR
                        ;38017 ;
                        ;38018 ;       Input           MEMSCAR         Address of desired MEMSCR
                        ;38019 ;                       PL              24(DEC)
                        ;38020 ;                       SL              4
                        ;38021 ;
                        ;38022 ;       Resources       MTEMP9          Temporary
                        ;38023 ;
                        ;38024 ;       Subroutines     MS.DEC.SP       VA <- SP <- SP - 4
                        ;38025 ;
                        ;38026 ;       Zero extend the selected memory status and control register and
                        ;38027 ;       push it on the stack.
                        ;38028 ;*****************************************************************************
                        ;38029
                        ;38030 =0
                        ;38031 IE.WRITE.MSCR:
                        ;38032         ;0------------------------------; LOAD MEMORY STATUS AND CONTROL
                        ;38033         M[TEMP9] MEMSCR,                ; REGISTER ADDRESSED BY MSCAR
                        ;38034         SET STACK FLAG,                 ;
U 0F30, 046E,9036,4030,0647,04CB,3      ;38035         PUSH,NEXT/MS.DEC.SP             ; PREPARE TO PUSH ON THE STACK
                        ;38036
                        ;38037 IE.WRITE.MSCR.0:
                        ;38038         ;1------------------------------; PL MUST BE 24 AND SL MUST BE 4 TO
                        ;38039         WRITE M[TEMP9].XZ,SIZE[LONG],   ; EXTRACT THE 4-BIT REGISTER, AND PUSH
U 0F31, 0028,9077,00A0,05D8,0000,1      ;38040         CLEAR STACK FLAG,RETURN [+1]    ; IT ON THE STACK.
```

```
                        ;38041 .TOC "  Interrupts and Exceptions       : INTERVAL TIMER INTERRUPT"
                        ;38042
                        ;38043 ;*****************************************************************************
                        ;38044 ;       Entry points    IE.INT.TIMER
                        ;38045 ;
                        ;38046 ;       Output          MTEMP8          New PSL
                        ;38047 ;                       VA              Interrupt vector address
                        ;38048 ;                       FLAG2           Set to indicate no more params
                        ;38049 ;
                        ;38050 ;       Resources       MTEMP8          Save TCSR.IICR
                        ;38051 ;                       TCSR.IICR       Used to reset the interrupt
                        ;38052 ;
                        ;38053 ;       Subroutines     IE.INTERRUPT
                        ;38054 ;*****************************************************************************
                        ;38055
                        ;38056 ;IE.INT.TIMER:
                        ;38057 ;       ;-------------------------------;
                        ;38058 ;       VA_M[SCBB]+ZLIT0[0C0],          ; LOAD THE MACRO VECTOR ADDRESS.
                        ;38059 ;       SET FLAG2,NEXT/IE.INT.TIMER2    ; NO MORE PARAMETERS TO PUSH.
                        ;38060
                        ;38061 0F6E:   ;*********FORCE ADDRESS*********;
                        ;38062 IE.INT.TIMER2:
                        ;38063         ;-------------------------------;
                        ;38064         M[TEMP8],ZLIT16[18],            ; SET IPL=18 IN NEW PSL.
U 0F6E, 0586,8D37,0030,C047,00EE,E      ;38065         NEXT/IE.INTERRUPT               ; INITIATE THE INTERRUPT.
```

```
                        ;38066 .TOC "  Interrupts and Exceptions       : TU58 INTERRUPT"
                        ;38067
                        ;38068 ;*****************************************************************************
                        ;38069 ;       Entry points    IE.TU58
                        ;38070 ;
                        ;38071 ;       Output          VA      Interrupt vector address
                        ;38072 ;
                        ;38073 ;       Resources       PL      Used for rotating TU58 registers.
                        ;38074 ;                       TEMP5   Temp to hold TU58 registers.
                        ;38075 ;
                        ;38076 ;       TU58 interrupts are signaled by the hardware like a unibus interrupt
                        ;38077 ;       without a write vector occuring.  If a unibus interrupt is detected
                        ;38078 ;       and a write vector did not occur control is passed to here from the
                        ;38079 ;       unibus interrupt flows.
                        ;38080 ;*****************************************************************************
                        ;38081
                        ;38082 IE.TU58.CONT:
                        ;38083          ;-------------------------------;
U 0F78, 0986,5EF6,4030,B3E7,00F7,C      ;38084          M[TEMP5]_TU58REGS,PL_[22.]      ; FETCH TU58 RECEIVER C/S REGISTER
                        ;38085
                        ;38086          ;-------------------------------;
                        ;38087          WB_.NOT.(M[TEMP5].RR.P),        ; CHECK TU58 INTERRUPT ENABLE AND
U 0F7C, 0C80,5176,03B0,0047,00F3,4      ;38088          WB<1-0>.NE.0?                   ; RECEIVER DONE BITS.
                        ;38089
                        ;38090 =0       ;0-------------------------------; TU58 RECEIVER INTERRUPT.
                        ;38091          VA_M[SCBB]+ZLIT0[0F0],          ; LOAD RECEIVER VECTOR ADDRESS.
U 0F34, 0580,EC11,0037,84A7,00F9,4      ;38092          NEXT/IE.UNIBUS.INT5             ; INITIATE THE INTERRUPT.
                        ;38093
                        ;38094          ;1-------------------------------; CHECK IF  TU58 TRANSMITTER
U 0F35, 0980,0D37,0036,0267,00F7,E      ;38095          TRAR_ZLIT16[0C0]                ; LOAD ADDRESS OF TU58 C/S REGISTER
                        ;38096
                        ;38097          ;-------------------------------;
U 0F7E, 0C86,5036,4030,03E7,00F8,3      ;38098          M[TEMP5]_TU58REGS               ; FETCH TU58 C/S REGISTER
                        ;38099
                        ;38100          ;-------------------------------;
                        ;38101          WB_M[TEMP5].RR.P,               ; CHECK TU58 XMTR INTERRUPT PENDING
U 0F83, 0480,5177,06F0,0047,00F1,D      ;38102          WB<31-30>?                      ; BY ROTATING IT TO BIT 31.
                        ;38103
                        ;38104 =01      ;01------------------------------;
                        ;38105          PC_M[PCBACK]-ZLIT0[2],          ; NOT A REAL INTERRUPT, CONTINUE MACRO.
U 0F1D, 0D81,9C10,0030,1487,0000,2      ;38106          NEXT/GL.NOP.IRD1                ; CAN'T LOAD PC AND DO IRD1 IN SAME CYCL
                        ;38107
                        ;38108          ;11------------------------------; TU58 TRANSMITTER INTERRUPT
U 0F1F, 0D80,0D37,0031,02E7,00F8,A      ;38109          TU58REGS_ZLIT16[20]             ; CLEAR TU58 TRANSMITTER INTERRUPT
                        ;38110
                        ;38111          ;-------------------------------; TU58 TRANSMITTER INTERRUPT.
                        ;38112          VA_M[SCBB]+ZLIT0[0F4],          ; LOAD TRANSMITTER VECTOR ADDRESS.
U 0F8A, 0D80,EC11,0037,A4A7,00F9,4      ;38113          NEXT/IE.UNIBUS.INT5             ; INITIATE THE INTERRUPT.
```

```
                                ;38114 .TOC "  Interrupts and Exceptions       : UNIBUS INTERRUPT"
                                ;38115
                                ;38116 ;*************************************************************************
                                ;38117 ;       Entry points    IE.UNIBUS.INT
                                ;38118 ;
                                ;38119 ;       Output          MTEMP8          New PSL
                                ;38120 ;                       VA              Interrupt vector address
                                ;38121 ;                       FLAG2           Set to indicate no more params
                                ;38122 ;
                                ;38123 ;       Resources       TEMP7           Temporary
                                ;38124 ;                       PC              Modified if not a real interrupt
                                ;38125 ;
                                ;38126 ;       Subroutines     IE.INTERRUPT
                                ;38127 ;
                                ;38128 ;       Issue a UNIBUS grant then see if there is an interrupt vector in
                                ;38129 ;       the MDR by checking MSCR #2.
                                ;38130 ;       If a write vector has not occured then it may be a TU58 interrupt.
                                ;38131 ;       Pass control to the TU58 interrupt code.  If there is also no
                                ;38132 ;       TU58 interrupt then point the PC back to the opcode of the
                                ;38133 ;       interrupted instruction and continue.
                                ;38134 ;       If a write vector has occured then load the interrupt vector address
                                ;38135 ;       into VA, load the IPL bits in the new PSL and jump to IE.INTERRUPT.
                                ;38136 ;*************************************************************************
                                ;38137
                                ;38138 ;IE.UNIBUS.INT:
                                ;38139 ;       ;--------------------------------;
                                ;38140 ;       MEMSCAR_ZLIT24[2],              ; LOAD ADDRESS OF UNIBUS INTERRUPT REG.
                                ;38141 ;       NEXT/IE.UNIBUS.INT2             ; JUMP TO UNIBUS INTERRUPT ROUTINE.
                                ;38142
                                ;38143 0F77:   ;*********FORCE ADDRESS*********;
                                ;38144 IE.UNIBUS.INT2:
                                ;38145         ;--------------------------------; ISSUE THE UNIBUS GRANT
U 0F77, 0086,8036,4030,066F,00F9,0 ;38146      BUS GRANT M[TEMP8]_IPL          ; AND SAVE THE IPL IN MTMP8.
                                ;38147
                                ;38148         ;--------------------------------;
U 0F90, 0886,7036,4030,0647,00F9,1 ;38149      M[TEMP7]_MEMSCR                 ; READ UNIBUS INTERRUPT REG.
                                ;38150
                                ;38151         ;--------------------------------;
U 0F91, 0980,7C92,0A70,0847,00F3,C ;38152      M[TEMP7] BIT24?                 ; SEE IF THERE IS AN INTERRUPT.
```

```
                                ;38153 =0
                                ;38154 IE.TU58:
                                ;38155          ;0------------------------------;  CHECK FI TU58 RECEIVER INTERRUPT
                                ;38156          TRAR ZLIT16[80],SET FLAG2,      ;  LOAD ADDRESS OF TU58 RECEIVER C/S
U 0F3C, 0550,0D37,0034,0267,00F7,8 ;38157       NEXT/IE.TU58.CONT               ;  JUMP TO TU58 FLOWS.
                                ;38158
                                ;38159          ;1------------------------------;
U 0F3D, 0080,05B7,0030,0607,00F9,2 ;38160       MEMSCR_0                        ;  CLEAR BIT FOR READ LOCK TIMEOUT
                                ;38161
                                ;38162          ;------------------------------;
                                ;38163          R[TEMP7] M[SCBB],               ;  MOVE SCBB TO AN RTEMP
U 0F92, 0C54,E592,4031,C047,00F9,3 ;38164       SET FLAG2                       ;  NO MORE PARAMETERS TO PUSH.
                                ;38165
                                ;38166          ;------------------------------;
                                ;38167          VA_ZEXT(M[MDR])+R[TEMP7],       ;  COMPUTE MACRO VECTOR ADDRESS.
                                ;38168          SIZE[WORD],                     ;
U 0F93, 0C81,2015,0011,C4A7,00F9,4 ;38169       NEXT/IE.UNIBUS.INT5             ;  INITIATE THE INTERRUPT.
                                ;38170
                                ;38171 IE.UNIBUS.INT5:
                                ;38172          ;------------------------------;
                                ;38173          M[TEMP8] MB.AND.ZLIT16[1F],     ;  CLEAR NON IPL BITS IN THE NEW PSL.
U 0F94, 0986,8D12,0030,F847,00EE,E ;38174       NEXT/IE.INTERRUPT               ;
```

```
                                ;38175 .TOC "  Interrupts and Exceptions       : CONSOLE INTERRUPT"
                                ;38176
                                ;38177 ;*************************************************************************
                                ;38178 ;       Entry points    IE.CONSOLE.INT
                                ;38179 ;
                                ;38180 ;       Output          MTEMP8          New PSL
                                ;38181 ;                       VA              Interrupt vector address
                                ;38182 ;                       FLAG2           Set to indicate no more params
                                ;38183 ;
                                ;38184 ;       Resources       CRAR
                                ;38185 ;                       TEMP5           Temporary
                                ;38186 ;
                                ;38187 ;       Subroutines     IE.INTERRUPT
                                ;38188 ;
                                ;38189 ;       If there is a console receiver interrupt it is serviced first, if
                                ;38190 ;       there is not a console receiver interrupt it is ASSUMED that there
                                ;38191 ;       is a console transmitter interrupt.
                                ;38192 ;*************************************************************************
                                ;38193
                                ;38194 ;IE.CONSOLE.INT:
                                ;38195 ;       ;-------------------------------;
                                ;38196 ;       M[TEMP8]_ZLIT16[14],            ; LOAD THE NEW IPL INTO THE NEW PSL.
                                ;38197 ;       NEXT/IE.CONSOLE.INT2            ; JUMP TO CONSOLE INTERRUPT ROUTINE.
                                ;38198
                                ;38199 0F9C:   ;*********FORCE ADDRESS*********;
                                ;38200 IE.CONSOLE.INT2:
                                ;38201         ;-------------------------------; LOAD ADDRESS OF CONSOLE RECEIVER
U 0F9C, 0580,0D37,0034,0247,00F9,5 ;38202      CRAR_ZLIT16[80]                 ; C/S REGISTER.
                                ;38203
                                ;38204         ;-------------------------------;
                                ;38205         M[TEMP5]_CONREGS,PL_[22.],      ; FETCH CONSOLE RECEIVER C/S REGISTER
U 0F95, 0156,5EF6,4030,B3C7,00F9,6 ;38206      SET FLAG2                       ; NO MORE PARAMETERS TO PUSH.
                                ;38207
                                ;38208         ;-------------------------------;
                                ;38209         WB_.NOT.(M[TEMP5].RR.P),        ; CHECK CONSOLE INTERRUPT ENABLE AND
U 0F96, 0480,5176,03B0,0047,00F4,4 ;38210      WB<1-0>.NE.0?                   ; RECEIVER DONE BITS.
                                ;38211
                                ;38212 =0      ;0------------------------------; CONSOLE RECEIVER INTERRUPT.
                                ;38213         VA_M[SCBB]+ZLIT0[0F8],          ; LOAD RECEIVE VECTOR ADDRESS.
U 0F44, 0180,EC11,0037,C4A7,00EE,E ;38214      NEXT/IE.INTERRUPT               ; INITIATE THE INTERRUPT.
                                ;38215
                                ;38216         ;1------------------------------; CONSOLE TRANSMITTER INTERRUPT
U 0F45, 0980,0D37,0036,0247,00F9,7 ;38217      CRAR_ZLIT16[0C0]                ;LOAD ADDR OF CONSOLE CONTROL AND STATUS
```

```
                                        ;38218          ;--------------------------------;
U 0F97, 0486,5036,403C,03C7,00F9,8      ;38219          M[TEMP5]_CONREGS                 ; READ CONSOLE CONTROL AND STATUS
                                        ;38220
                                        ;38221          ;--------------------------------;
U 0F98, 0986,5D13,8032,0047,00F9,9      ;38222          M[TEMP5]_MB.ANDNOT.ZLIT16[40]    ; DON'T CLEAR HALT WHEN CLEARING XMIT
                                        ;38223
                                        ;38224          ;--------------------------------; CLEAR CONSOLE TRANSMITTER INTERRUPT
U 0F99, 0D80,5D12,4031,02C7,00F9,A      ;38225          CONREGS_M[TEMP5].OR.ZLIT16[20]   ;
                                        ;38226
                                        ;38227          ;--------------------------------; CONSOLE TRANSMITTER INTERRUPT.
                                        ;38228          VA_M[SCBB]+ZLIT0[0FC],           ; LOAD TRANSMITTER VECTOR ADDRESS.
U 0F9A, 0980,EC11,0037,E4A7,00EE,E      ;38229          NEXT/IE.INTERRUPT                ; INITIATE THE INTERRUPT.
```

```
                                ;38230 .TOC "  Interrupts and Exceptions       : KERNAL STACK NOT VALID"
                                ;38231
                                ;38232 ;*****************************************************************************
                                ;38233 ;       Entry points    IE.KSNV
                                ;38234 ;
                                ;38235 ;       Input           PSL<IS>         0 means KSNV, 1 means ISNV
                                ;38236 ;                       TEMP4           PSL that would've been pushed if no KSNV
                                ;38237 ;
                                ;38238 ;       Output          PSL             PSL that would've been pushed if no KSNV
                                ;38239 ;                       FLAG0           Clear to indicate an exception
                                ;38240 ;                       FLAG1           Clear to indicate on interrupt stack
                                ;38241 ;                       FLAG2           Set to indicate no more params to push
                                ;38242 ;                       TEMP5           Interrupt vector
                                ;38243 ;                       FPDOFFSET       Set to 4 if ISNV
                                ;38244 ;
                                ;38245 ;       Resources       MTEMP9          Temporary to switch stacks
                                ;38246 ;                       STACK FLAG
                                ;38247 ;                       MM NOINT
                                ;38248 ;
                                ;38249 ;       Subroutines     IE.50           An entry in IE.INTERRUPT
                                ;38250 ;
                                ;38251 ;       Check PSL<IS> if it is set then halt, otherwise switch to the kernal
                                ;38252 ;       stack, fetch the interrupt vector, restore the old PSL and jump
                                ;38253 ;       to IE.50 to finish initiating the exception.
                                ;38254 ;*****************************************************************************
                                ;38255
                                ;38256 IE.KSNV:
                                ;38257          ;-------------------------------;
                                ;38258          SET MM.NOINT,                   ; TURN OFF MEMORY MANAGEMENT INTERRUPTS.
U 0F9B, 0C60,0036,47B0,0207,08F0,B 479*         ;38259          PSL<IS.CURM>?                   ; CHECK PSL<IS> IN NEW PSL.
                                ;38260
                                ;38261 =1011    ;1011---------------------------; NEW PSL<IS>=0 MUST BE KSNV.
                                ;38262          R[KSP]_M[TEMP9],                ; SAVE OLD STACK POINTER IN KSP.
                                ;38263          CLEAR FLAG1,                    ; CLEAR TO FORCE IPL TO 1F.
U 0F0B, 080C,9592,4038,0047,00F9,E              ;38264          NEXT/IE.KSNV.10                 ;
                                ;38265
                                ;38266          ;1111---------------------------; NEW PSL<IS>=1 MUST BE ISNV.
U 0F0F, 0480,4002,403D,8007,00F9,D              ;38267          PSL_M[TEMP4]                    ; RESTORE OLD PSL.
                                ;38268
                                ;38269          ;-------------------------------;
                                ;38270          M[FPDOFFSET]_#LIT0[4],          ; SET ERROR CODE
                                ;38271          SET STACK FLAG,                 ; TELL CONSOLE TO LOOK AT FRONT PANEL
U 0F9D, 0D6E,CC37,0030,2047,0000,1              ;38272          NEXT/MS.HALT.MICRO              ;
```

```
                                ;38273 IE.KSNV.10:
                                ;38274          ;--------------------------------;
U 0F9E, 0486,95BE,4039,0047,00F9,F ;38275       M[TEMP9]_R[ISP]                  ; PREPARE TO LOAD NEW STACK POINTER.
                                ;38276
                                ;38277          ;--------------------------------;
                                ;38278          VA_M[SCBB]+ZLITO[8],             ; LOAD ADDRESS OF MACRO VECTOR.
U 0F9F, 0150,EC11,0030,44A7,00FA,0 ;38279       SET FLAG2                        ; NO MORE PARAMETERS TO PUSH.
                                ;38280
                                ;38281          ;--------------------------------; SET IS=1 TO TRAP FAILURE OF READ.PHY
                                ;38282          PSL(PREV_CURM ISCURM_[4]),       ; WHICH WILL FORCE AN ISNV.
U 0FA0, 0168,0CB7,0030,26A7,00FA,1 ;38283       SET STACK FLAG                   ;
                                ;38284
                                ;38285          ;--------------------------------; READ THE MACRO VECTOR AND
U 0FA1, 0C00,0036,4030,0040,00FA,2 ;38286       READ.PHY,CLEAR FLAG0             ; CLEAR FLAG TO INDICATE AN EXCEPTION.
                                ;38287
                                ;38288          ;--------------------------------;
U 0FA2, 0C80,4002,403D,8007,00FA,3 ;38289       PSL_M[TEMP4]                     ; RESTORE OLD PSL.
                                ;38290
                                ;38291          ;--------------------------------;
                                ;38292          R[TEMP5]_M[MDR],CLEAR TP,        ; SAVE THE MACRO VECTOR.
U 0FA3, 0C95,2592,4271,4047,00F3,9 ;38293       WB<1-0>?                         ;
                                ;38294
                                ;38295 =01      ;01-------------------------------; VECTOR<1:0> = 1 OR 0 HANDLE ON IS.
                                ;38296          R[SP]_M[TEMP9],                  ; LOAD NEW STACK POINTER FROM IS.
                                ;38297          CLEAR STACK FLAG,                ;
U 0F39, 0C2C,9592,4037,8047,00ED,F ;38298       NEXT/IE.50                       ; BRANCH TO END OF INITIATE ROUTINE.
                                ;38299
                                ;38300          ;11-------------------------------; VECTOR<1:0> = 2 OR 3
                                ;38301          R[SP]_M[TEMP9],                  ; LOAD NEW STACK POINTER FROM IS.
                                ;38302          CLEAR STACK FLAG,                ;
U 0F3B, 0C2C,9592,4037,8047,00EF,2 ;38303       NEXT/IE.WCS.BASE                 ; HANDLE IN WCS.
```

```
                                ;38304 .TOC "  Interrupts and Exceptions       : ACV AND TNV FAULTS"
                                ;38305
                                ;38306 ;*****************************************************************************
                                ;38307 ;       Entry points    IE.ACV.READ
                                ;38308 ;                       IE.ACV.WRITE
                                ;38309 ;                       IE.ACV.XB
                                ;38310 ;                       IE.ACV
                                ;38311 ;                       IE.ACV.BUT
                                ;38312 ;                       IE.ACV.RBACK
                                ;38313 ;                       IE.TNV.RBACK
                                ;38314 ;                       IE.TNV
                                ;38315 ;
                                ;38316 ;       Input           ERRCOD          Error code (unless utrap entry)
                                ;38317 ;                       MM.TEMP3        Faulting address (only utrap entry)
                                ;38318 ;
                                ;38319 ;       Output          VA              Interrupt vector address
                                ;38320 ;                       FLAG2           Clear to indicate more params to push
                                ;38321 ;
                                ;38322 ;       Resources       MM.TEMP3        Save faulting address on utrap entry
                                ;38323 ;                       FLAG1           Indicates if on interrupt stack
                                ;38324 ;                       D               Temporary
                                ;38325 ;                       TEMP0           Save RBSP when rolling back
                                ;38326 ;                       TEMP6           Save faulting address
                                ;38327 ;
                                ;38328 ;       Subroutines     IE.RBS.RBACK    FPD=0 roll back
                                ;38329 ;                       IE.FPD.PACK     FPD=1 pack it up
                                ;38330 ;                       IE.FAULT        Initiate the fault
                                ;38331 ;                       MS.DEC.SP       VA <- SP <- (SP - 4)
                                ;38332 ;                       IE.LOAD.PC+1    Finish initiating the exception
                                ;38333 ;
                                ;36334 ;       There are micro code detected ACV/TNV's and hardware detected
                                ;38335 ;       ACV/TNV's which take different points.  The hardware utrap entries
                                ;38336 ;       must first set the error code and save the faulting address.
                                ;38337 ;       Then they all load the macro vector address and either roll back
                                ;38338 ;       register side effects or do FPD packing.  Then IE.FAULT is called
                                ;38339 ;       and then the error code and faulting address are pushed on the stack
                                ;38340 ;       and IE.LOAD.PC+1 is called to finish initiating the exception.
                                ;38341 ;*****************************************************************************
                                ;38342 002F:   ;**UTRAP ADDR**                  ;ENTRY FOR MICRO TRAP DETECTED ACV WRITE
                                ;38343 IE.ACV.WRITE:
                                ;38344         ;-------------------------------;
                                ;38345         M[ERRCOD]_ZLIT0[4],              ; SET ERROR CODE (SRM 5.5.4)
U 002F, 0186,BC37,0030,2047,0002,B ;38346      NEXT/MM.WRITE.TBMISS             ;
                                ;38347
                                ;38348 002E:   ;**UTRAP ADDR**                  ; ENTRY FOR MICRO TRAP DETECTED ACV READ
                                ;38349 IE.ACV.READ:
                                ;38350         ;-------------------------------;
U 002E, 0186,BC37,0030,0047,0037,6 ;38351      M[ERRCOD]_ZLIT0[0]               ; SET ERROR CODE (SRM 5.5.4)
                                ;38352
                                ;38353 0376:   ;*********FORCE ADDRESS*********;
U 0376, 0885,2592,4039,8047,00EF,F ;38354      R[MM.TEMP2]_M[MDR]               ; SAVE MDR
                                ;38355
                                ;38356 0EFF:   ;*********FORCE ADDRESS*********;
                                ;38357         R[MM.TEMP3]_M[VA],               ; SAVE FAULTING ADDRESS
U 0EFF, 0C85,B592,4039,C047,0159,3 ;38358      NEXT/MM.READ.NO.ACCESS           ;
```

```
                                        ;38359 0FAE:   ;*********FORCE ADDRESS*********;
                                        ;38360 FREE.0FAE:
                                        ;38361         ;-------------------------------; GET READ/MODIFY BIT TO SEPERATE
U 0FAE, 0884,0036,4033,C647,00FA,4      ;38362         R[MM.TEMP1]_MEMSCR              ; READ/MODIFY FROM STRAIGHT READS.
                                        ;38363
                                        ;38364 IE.READ.ACV:
                                        ;38365         ;-------------------------------;
                                        ;38366         WB_R[MM.TEMP1].RR.24,           ; BRANCH ON READ/MODIFY BIT.
U 0FA4, 0C80,02B7,0223,C047,0977,7 337* ;38367         WB<5-0>?                        ;
                                        ;38368
                                        ;38369 1777:   ;*********FORCE ADDRESS*********;
U 1777, 0986,BC12,4030,2047,0177,F      ;38370         M[ERRCOD]_MB.OR.ZLIT0[4]        ; TURN ON WRITE/MODIFY BIT IN ERROR CODE
                                        ;38371
                                        ;38372 177F:   ;*********FORCE ADDRESS*********;
                                        ;38373         MDR_R[MM.TEMP2],                ; RESTORE MDR FOR PACKING
U 177F, 0880,05BE,4039,8467,00FB,0      ;38374         NEXT/IE.ACV.CHK.CONSOLE         ;
                                        ;38375 0EF7:
                                        ;38376 FREE.0EF7:
                                        ;38377         ;*********FORCE ADDRESS*********;
                                        ;38378         ;110111-----------------------;
U 0EF7, 0986,BC12,4030,2047,0002,3      ;38379         M[ERRCOD]_MB.OR.ZLIT0[4]        ; TURN ON WRITE/MODIFY BIT IN ERROR CODE
                                        ;38380
                                        ;38381 0023:   ;**UTRAP ADDR**                 ;ENTRY FOR MICRO TRAP DETECTED XB ACV
                                        ;38382 IE.ACV.XB:
                                        ;38383         ;-------------------------------;
                                        ;38384         M[ERRCOD]_ZLIT0[0],             ; SET ERROR CODE (SRM 5.5.4)
U 0023, 0986,BC37,0030,0047,0002,2      ;38385         NEXT/MM.XB.TBMISS               ;
                                        ;38386
                                        ;38387 0F2E:   ;*********FORCE ADDRESS*********;
                                        ;38388         ;1110-------------------------; NO ACCESS (DIDN'T FAULT ACROSS A PB).
                                        ;38389         D_M[SCBB]+ZLIT0[20],            ; LOAD MACRO VECTOR ADDRESS.
U 0F2E, 0D80,EC11,23F1,0047,00F5,0      ;38390         FPD?,NEXT/IE.PACK.OR.RBACK      ; BRANCH ON FPD.
                                        ;38391
                                        ;38392 0F2F:   ;1111-------------------------; ACCESS OK (FAULTED ACROSS A PB).
                                        ;38393         R[MM.TEMP3]_D+RB,               ; INCREMENT ADDRESS INTO FAULTING PAGE.
U 0F2F, 0C84,0021,0039,C047,00F5,D      ;38394         NEXT/IE.ACV                     ;
                                        ;38395
                                        ;38396 0FB0:   ;*********FORCE ADDRESS*********;
                                        ;38397 IE.ACV.CHK.CONSOLE:
                                        ;38398         ;-------------------------------;
                                        ;38399         WB_M[FPDOFFSET],                ; CHECK IF IN CONSOLE MODE
U 0FB0, 0880,C592,46F0,0047,00F5,D      ;38400         WB<31-30>?,NEXT/IE.ACV          ;
```

```
                                ;38401 0F5D:    ;*********FORCE ADDRESS*********;
                                ;38402 IE.ACV:
                                ;38403          ;01-----------------------------;
                                ;38404          D_M[SCBB]+ZLIT0[20],            ; LOAD MACRO VECTOR ADDRESS.
U 0F5D, 0D80,EC11,23F1,0047,00F5,0 ;38405       FPD?,NEXT/IE.PACK.OR.RBACK      ; BRANCH ON FPD.
                                ;38406
                                ;38407 0F5F:    ;11-----------------------------; EXAMINE/DEPOSIT FROM THE CONSOLE
                                ;38408          WB_M[FPDOFFSET],WB<5-0>?,       ; RETURN TO CONSOLE OR BOOT.
U 0F5F, 0480,C592,4230,0047,00EF,8 ;38409       NEXT/IE.RET.TO.CN.BT.MV         ;
                                ;38410
                                ;38411 002D:    ;**UTRAP ADDR**                 ; ENTRY FOR UTRAP DETECTED BUT XB ACV
                                ;38412 IE.ACV.BUT:
                                ;38413          ;-----------------------------;
U 002D, 0880,0036,4830,0047,00F3,2 ;38414       OTHER UTRAPS?,NEXT/IE.ACV.BUT.1 ; CHECK IF OTHER HIGHER PRIORITY UTRAPS.
                                ;38415
                                ;38416 0F20:    ;*********FORCE ADDRESS*********;
                                ;38417          ;0**00-----------------------;  IE.ACV.SAV.FA DOES RETURN-13
                                ;38418          PROBE READ?,SIZE[BYTE],         ; CHECK FOR FAULT ACROSS A PAGE BOUNDARY
U 0F20, 0980,0C37,2780,205F,08F4,E 479* ;38419  D_ZLIT0[4],NEXT/IE.ACV.RBACK    ; LOAD 4 INTO D TO USE AS AN INCREMENT.
                                ;38420
                                ;38421 0F32:    ;*********FORCE ADDRESS*********;
                                ;38422 IE.ACV.BUT.1:
                                ;38423          ;1**10-----------------------;
U 0F32, 0C80,0036,4030,0047,015F,C ;38424       NEXT/MM.GET.BUT.PTE             ; SOME OTHER UTRAP, DO IT FIRST.
                                ;38425
                                ;38426 0F33:    ;1**11-----------------------;
                                ;38427          M[ERRCOD]_ZLIT0[0],             ; LOAD ERROR CODE (SRM 5.5.4)
U 0F33, 0986,BC37,0030,0047,04FB,6 ;38428       PUSH,NEXT/IE.ACV.SAV.FA         ; SAVE THE FAULTING ADDRESS AND LOAD VA.
                                ;38429
                                ;38430 0F4E:    ;*********FORCE ADDRESS*********;
                                ;38431 IE.ACV.RBACK:
                                ;38432          ;1110-----------------------;  NO ACCESS (DIDN'T FAULT ACROSS A PB).
                                ;38433          D_M[SCBB]+ZLIT0[20],            ; LOAD MACRO VECTOR ADDRESS.
U 0F4E, 0D80,EC11,2031,0047,00F5,0 ;38434       NEXT/IE.PACK.OR.RBACK           ; ROLL BACK RBS (IGNORE FPD).
                                ;38435
                                ;38436 0F4F:    ;1111-----------------------;  ACCESS OK (FAULTED ACROSS A PB).
                                ;38437          R[MM.TEMP3]_D+RB,               ; INCREMENT ADDRESS INTO FAULTING PAGE.
U 0F4F, 0484,0021,0039,C047,00F4,E ;38438       NEXT/IE.ACV.RBACK               ; ROLL BACK RBS (IGNORE FPD).
                                ;38439
                                ;38440 0FB1:    ;*********FORCE ADDRESS*********;
                                ;38441 IE.TNV.RBACK:
                                ;38442          ;-----------------------------;
                                ;38443          D_M[SCBB]+ZLIT0[24],            ; LOAD MACRO VECTOR ADDRESS.
U 0FB1, 0580,EC11,2031,2047,00F5,0 ;38444       NEXT/IE.PACK.OR.RBACK           ; ROLL BACK RBS (IGNORE FPD).
                                ;38445
                                ;38446 0FB2:    ;*********FORCE ADDRESS*********;
                                ;38447 FREE.0FB2:
                                ;38448          ;-----------------------------; GET READ/MODIFY BIT TO SEPERATE
U 0FB2, 0484,0036,4033,8647,00FA,5 ;38449       R[MM.TEMP5]_MEMSCR              ; READ/MODIFY FROM STRAIGHT READS.
```

```
                                        ;38450 IE.READ.TNV:
                                        ;38451          ;--------------------------------;
                                        ;38452          WB_R[MM.TEMP5].RR.24,           ; BRANCH ON READ/MODIFY BIT.
U 0FA5, 0080,02B7,0223,8047,08F3,7 337* ;38453          WB<5-0>?                        ;
                                        ;38454 =110111
                                        ;38455          ;110111--------------------------;
                                        ;38456          MDR_R[MM.TEMP2],                ; THIS WAS A READ/MODIFY, REDO AS A
U 0F37, 0080,05BE,4039,8467,015A,1      ;38457          NEXT/MM.WRITE.TBMISS.10          ; WRITE TBMISS
                                        ;38458
                                        ;38459          ;111111--------------------------; THIS WAS NOT A READ/MODIFY,
                                        ;38460          MDR_R[MM.TEMP2],                ; RESTORE MDR FOR PACKING
U 0F3F, 0880,05BE,4039,8467,00FB,5      ;38461          NEXT/IE.TNV.CHK.CONSOLE          ; TAKE A TNV FAULT
                                        ;38462
                                        ;38463 0FB5:    ;*********FORCE ADDRESS*********;
                                        ;38464 IE.TNV.CHK.CONSOLE:
                                        ;38465          ;--------------------------------;
                                        ;38466          WB_M[FPDOFFSET],                ;
U 0FB5, 0880,C592,46F0,0047,00F6,1      ;38467          WB<31-30>?,NEXT/IE.TNV          ; CHECK IF IN CONSOLE MODE
                                        ;38468
                                        ;38469 0F61:    ;*********FORCE ADDRESS*********;
                                        ;38470 IE.TNV:
                                        ;38471          ;01-------------------------------;
                                        ;38472          D_M[SCBB]+ZLIT0[24],            ; LOAD MACRO VECTOR ADDRESS.
U 0F61, 0580,EC11,23F1,2047,00F5,0      ;38473          FPD?,NEXT/IE.PACK.OR.RBACK      ; BRANCH ON FPD.
                                        ;38474
                                        ;38475 0F63:    ;11-------------------------------; EXAMINE/DEPOSIT FROM THE CONSOLE
                                        ;38476          WB_M[FPDOFFSET],WB<5-0>?,       ; RETURN TO CONSOLE OR BOOT.
U 0F63, 0480,C,92,4230,0047,00EF,8      ;38477          NEXT/IE.RET.TO.CN.BT.MV         ;
                                        ;38478
                                        ;38479 =0
                                        ;38480 IE.ACV.PUSH.PARAM:
                                        ;38481          ;0--------------------------------;
                                        ;38482          WRITE M[TEMP6],SIZE[LONG],      ; PUSH THE FAULTING ADDRESS.
U 0F48, 0480,6592,4020,05D8,04CB,3      ;38483          PUSH,NEXT/MS.DEC.SP             ; PREPARE TO PUSH THE ERROR CODE.
                                        ;38484
                                        ;38485          ;1--------------------------------;
                                        ;38486          WRITE M[ERRCOD],SIZE[LONG],     ; PUSH THE ERROR CODE.
U 0F49, 0480,B592,4420,05D8,00EF,4      ;38487          FLAG0?,NEXT/IE.LOAD.PC          ; GO LOAD NEW PC AND PSL.
                                        ;38488
                                        ;38489 0FB6:    ;*********FORCE ADDRESS*********;
                                        ;38490 IE.ACV.SAV.FA:
                                        ;38491          ;--------------------------------;
                                        ;38492          VA_R[MM.TEMP3]_M[PC],           ; SAVE FAULTING ADDRESS.
U 0FB6, 0485,A592,40B9,C4A7,03FE,D      ;38493          RETURN [-13]                    ; ALSO LOAD FAULTING ADDRESS INTO VA.
```

```
;38494 .TOC "  Interrupts and Exceptions       : ARITH TRAPS AND TRACE PENDING FAULT" ==
;38495
;38496 ;*****************************************************************************
;38497 ;       Entry points    IE.ATCR
;38498 ;                       IE.FPTCR
;38499 ;                       IE.INT.DBZ
;38500 ;                       IE.FLT.DEC.DBZ
;38501 ;                       IE.FLT.OV
;38502 ;                       IE.INDEX.RANGE
;38503 ;                       IE.TP.FAULT
;38504 ;
;38505 ;       Output          VA              Macro vector address
;38506 ;                       FLAG2           Cleared to indicate more params to push
;38507 ;                       FLAG3           Set on micro code detected traps to
;38508 ;                                       indicate PC should be pushed on the
;38509 ;                                       stack.  It will be clear on hardware
;38510 ;                                       detected traps which will cause
;38511 ;                                       PCBACK-2 to be pushed.
;38512 ;
;38513 ;       Resources       ERRCOD          Save trap code
;38514 ;
;38515 ;       Subroutines     IE.TRAP
;38516 ;                       IE.FAULT
;38517 ;                       IE.LOAD.PC+1
;38518 ;
;38519 ;       Each entry loads the proper trap code into ERRCOD then loads the
;38520 ;       macro vector address into VA and calls IE.TRAP or IE.FAULT to
;38521 ;       initiate the exception, then the trap code is pushed and IE.LOAD.PC+1
;38522 ;       is called to finish initiating the exception.
;38523 ;*****************************************************************************
;38524
;38525 011:                                          ; FORCE ADDRESS FOR HARDWARE BRANCH.
;38526 IE.ATCR:
;38527                 ;-------------------------------;
;38528         M[ERRCOD]_ATCR,                         ; SAVE THE TRAP CODE.
U 0011, 0086,B036,4030,00E7,00F8,4      ;38529         NEXT/IE.ARITH.TRP                       ; JUMP TO COMMON FLOW.
;38530 012:                                          ; FORCE ADDRESS FOR HARDWARE BRANCH.
;38531 IE.FPTCR:
;38532                 ;-------------------------------;
;38533         M[ERRCOD]_FPTCR,WB<5-0>?,               ; BRANCH ON THE CODE FROM THE FPA
U 0012, 0486,B036,4230,0067,00FF,8      ;38534         NEXT/IE.FPA.FAULT.TRAP                  ;
;38535
;38536 0FB8:   ;*********FORCE ADDRESS*********;
;38537 IE.INT.DBZ:
;38538                 ;-------------------------------;
;38539         M[ERRCOD]_CONX(2),                      ; SAVE THE TRAP CODE.
;38540         SET V,                                  ; SET OVERFLOW.
U 0FB8, 0486,B737,0010,08E7,00FA,6      ;38541         NEXT/IE.CLEAR.TRP                       ; JUMP TO COMMON FLOW.
;38542
;38543 0FB9:   ;*********FORCE ADDRESS*********;
;38544 IE.FLT.DEC.DBZ:
;38545                 ;-------------------------------;
;38546         M[ERRCOD]_ZLIT0[4],                     ; SAVE THE TRAP CODE.
U 0FB9, 0186,BC37,0030,2047,00FA,6      ;38547         NEXT/IE.CLEAR.TRP                       ; JUMP TO COMMON FLOW.
```

```
                                ;38548 0FBA:    ;*********FORCE ADDRESS*********;
                                ;38549 IE.INDEX.RANGE:
                                ;38550          ;-------------------------------;
                                ;38551          M[ERRCOD]_ZLIT0[7],             ; SAVE THE TRAP CODE.
U 0FBA, 015E,BC37,0030,3847,00F8,4 ;38552       SET FLAG3,NEXT/IE.ARITH.TRP     ; JUMP TO COMMON FLOW.
                                ;38553
                                ;38554 IE.CLEAR.TRP:
                                ;38555          ;-------------------------------;
U 0FA6, 0458,0036,4030,00E7,00F8,4 ;38556       WB_ATCR,SET FLAG3               ; READ ATCR TO CLEAR TRAP.
                                ;38557
                                ;38558 0F84:    ;*********FORCE ADDRESS*********;
                                ;38559 IE.ARITH.TRP:
                                ;38560          ;0------------------------------;
                                ;38561          VA_M[SCBB]+ZLIT0[34],           ; LOAD MACRO VECTOR ADDRESS
                                ;38562          CLEAR FLAG2,                    ; MORE PARAMETERS TO PUSH.
U 0F84, 0510,EC11,0031,A4A7,04EC,C ;38563       PUSH,NEXT/IE.TRAP               ;
                                ;38564
                                ;38565 0F85:    ;1------------------------------;
U 0F85, 0C84,073C,0027,84A7,00FA,7 ;38566       VA_R[SP]_RB-CONX(4)             ; PREPARE TO PUSH
                                ;38567
                                ;38568 IE.ARITH.PSH.CODE:
                                ;38569          ;-------------------------------;
                                ;38570          WRITE M[ERRCOD].AND.ZLIT0[0F],  ; PUSH THE TRAP CODE.
                                ;38571          SIZE[LONG],                     ;
U 0FA7, 0580,BC12,0020,7DD8,00EF,4 ;38572       NEXT/IE.LOAD.PC                 ; GO LOAD NEW PC AND PSL.
                                ;38573
                                ;38574 15:                                      ; FORCE ADDRESS FOR HARDWARE BRANCH.
                                ;38575 IE.TP.FAULT:
                                ;38576          ;-------------------------------;
                                ;38577          VA_M[SCBB]+ZLIT0[28],           ; LOAD MACRO VECTOR ADDRESS
                                ;38578          SET FLAG2,                      ; NO MORE PARAMETERS TO PUSH.
U 0015, 0D50,EC11,0031,44A7,04EC,D ;38579       PUSH,NEXT/IE.FAULT              ;
```

```
                        ;38580 .TOC "  Interrupts and Exceptions       : COMPATABILITY MODE EXCEPTIONS"
                        ;38581
                        ;38582 ;*****************************************************************************
                        ;38583 ;       Entry points    IE.CM.RESOP
                        ;38584 ;                       IE.CM.BPT
                        ;38585 ;                       IE.CM.IOT
                        ;38586 ;                       IE.CM.EMT
                        ;38587 ;                       IE.CM.TRAP
                        ;38588 ;                       IE.CM.ILLINS
                        ;38589 ;                       IE.CM.ODD
                        ;38590 ;
                        ;38591 ;       Output          VA              Macro vector address
                        ;38592 ;                       FLAG2           Cleared to indicate more params to push
                        ;38593 ;
                        ;38594 ;       Resources       ERRCOD          Save trap code
                        ;36595 ;
                        ;38596 ;       Subroutines     IE.ABORT
                        ;38597 ;                       IE.LOAD.PC
                        ;38598 ;
                        ;38599 ;       Each entry loads the proper trap code into ERRCOD then loads the
                        ;38600 ;       macro vector address into VA and calls IE.ABORT to
                        ;38601 ;       initiate the exception, then the error code is pushed and IE.LOAD.PC
                        ;38602 ;       is called to finish initiating the exception.
                        ;38603 ;*****************************************************************************
                        ;38604
                        ;38605 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;38606 =1111
                        ;38607 IE.CM.RESOP:
                        ;38608          ;-------------------------------;
                        ;38609          M[ERRCOD] ZLIT0[0],             ; LOAD RESERVED OPERAND ERROR CODE.
U 010F, 0186,BC37,0030,0047,00FD,A      ;38610          NEXT/IE.CM.FAULT                ; JUMP TO COMMON FLOW.
                        ;38611 =111*
                        ;38612 IE.CM.BPT:
                        ;38613          ;-------------------------------;
                        ;38614          M[ERRCOD] CONX(1),              ; LOAD BREAK POINT ERROR CODE.
                        ;38615          MDR_ZEXT(OSR),SET FLAG0,        ; CHECK FOR RESERVED OPCODES
U 000E, 0C46,B737,0000,05E7,014D,0      ;38616          NEXT/CM-TEST.OSR                ;
                        ;38617 =111*
                        ;38618 IE.CM.IOT:
                        ;38619          ;-------------------------------;
                        ;38620          M[ERRCOD] CONX(2),              ; LOAD I/O TRAP ERROR CODE.
                        ;38621          MDR_ZEXT(OSR),SET FLAG0,        ; CHECK FOR RESERVED OPCODES
U 001E, 0846,B737,0010,05E7,014D,0      ;38622          NEXT/CM-TEST.OSR                ;
                        ;38623 =1111
                        ;38624 IE.CM.EMT:
                        ;38625          ;-------------------------------;
                        ;38626          M[ERRCOD] ZLIT0[3],             ; LOAD EMULATOR TRAP ERROR CODE.
U 011F, 0186,BC37,0030,1847,00FD,A      ;38627          NEXT/IE.CM.FAULT                ; JUMP TO COMMON FLOW.
                        ;38628 =1111
                        ;38629 IE.CM.TRAP:
                        ;38630          ;-------------------------------;
                        ;38631          M[ERRCOD] ZLIT0[4],             ; LOAD TRAP ERROR CODE.
U 012F, 0986,BC37,0030,2047,00FD,A      ;38632          NEXT/IE.CM.FAULT                ; JUMP TO COMMON FLOW.
                        ;38633 .REGION/IANDE.R1L,IANDE.R1H/IANDE.R2L,IANDE.R2H/IANDE.R3L,IANDE.R3H
```

```
                                ;38634 0FBE:    ;*********FORCE ADDRESS*********;
                                ;38635 IE.CM.ILLINS:
                                ;38636          ;-------------------------------;
                                ;38637          M[ERRCOD] ZLIT0[5],             ; LOAD ILLEGAL INSTRUCTION ERROR CODE.
U 0FBE, 0D86,BC37,0030,2847,00FD,A ;38638       NEXT/IE.CM.FAULT                ; JUMP TO COMMON FLOW.
                                ;38639
                                ;38640 0FD9:    ;*********FORCE ADDRESS*********;
                                ;38641 IE.CM.ODD:
                                ;38642          ;-------------------------------;
                                ;38643          M[ERRCOD] ZLIT0[6],             ; LOAD ODD ADDRESS ERROR CODE.
U 0FD9, 0D86,BC37,0030,3047,00F4,C ;38644       NEXT/IE.CM.EXCEP                ; JUMP TO COMMON FLOW.
                                ;38645
                                ;38646 0FDA:    ;*********FORCE ADDRESS*********;
                                ;38647 IE.CM.FAULT:
                                ;38648          ;-------------------------------;
U 0FDA, 0490,0036,4030,0047,00F4,C ;38649       CLEAR TP                        ; MUST CLEAR TP AFTER FIRST CYCLE
                                ;38650 =0
                                ;38651 IE.CM.EXCEP:
                                ;38652          ;0-------------------------------;
                                ;38653          VA_M[SCBB]+ZLIT0[30],           ; LOAD MACRO VECTOR ADDRESS.
                                ;38654          CLEAR FLAG2,                    ; MORE PARAMETERS TO PUSH.
U 0F4C, 0D10,EC11,0031,84A7,04EC,C ;38655       PUSH,NEXT/IE.ABORT              ;
                                ;38656
                                ;38657          ;1-------------------------------;
                                ;38658          VA_R[SP]_RB-CONX(4),            ;
U 0F4D, 0C6C,073C,0027,84A7,00FA,8 ;38659       SET STACK FLAG                  ; PREPARE TO PUSH.
                                ;38660
                                ;38661          ;-------------------------------;
                                ;38662          WRITE M[ERRCOD],SIZE[LONG],     ; PUSH THE ERROR CODE.
                                ;38663          CLEAR STACK FLAG,               ;
U 0FA8, 0C28,B592,4420,05D8,00EF,4 ;38664       FLAG0?,NEXT/IE.LOAD.PC          ; GO LOAD NEW PC AND PSL.
```

```
;38665 .TOC "  Interrupts and Exceptions       : ADDR MODE, RESV OPER, RESV OPCODE"
;38666 .TOC "  Interrupts and Exceptions       : FLOATING POINT FAULTS"
;38667
;38668 ;*************************************************************************
;38669 ;       Entry points    IE.ADDR.MODE    addressing mode fault
;38670 ;                       IE.OPCOD.DEC    opcode reserved to didital fault
;38671 ;                       IE.OPCOD.CUST   opcode reserved to customer fault
;38672 ;                       IE.OPER.FAULT   reserved operand fault
;38673 ;                       IE.OPER.ABORT   reserved operand abort
;38674 ;                       IE.FLOV.FAULT   floating overflow fault
;38675 ;                       IE.FLDBZ.FAULT  floating divide by zero fault
;38676 ;                       IE.FLUN.FAULT   floating underflow fault
;38677 ;
;38678 ;       Output          VA              Macro vector address
;38679 ;                       FLAG2           Set to indicate no more params to push
;38680 ;
;38681 ;       Resources       TEMP0           Save RBSP when rolling back
;38682 ;                       DREG            Save macro vector address
;38683 ;                       FPDOFFSET       Used if FPD=1 see IE.PACK.DONE
;38684 ;
;38685 ;       Subroutines     IE.RBS.RBACK    roll back register side effects
;38686 ;                       IE.ABORT
;38687 ;                       IE.FAULT
;38688 ;
;38689 ;       Save the macro vector address in DREG, then roll back the register side
;38690 ;       effects or pack up if FPD=1.  Finally move DREG (vector address) into
;38691 ;       VA and jump to IE.FAULT, for floating faults control will return and
;38692 ;       an error code will be pushed on the stack.
;38693 ;*************************************************************************
;38694
;38695 OFDC:   ;*********FORCE ADDRESS*********;
;38696 IE.ADDR.MODE:
;38697         ;-------------------------------;
;38698         D_M[SCBB]+ZLIT0[1C],            ; LOAD THE MACRO VECTOR ADDRESS.
U OFDC, 0150,EC11,2030,E047,00F6,4      ;38699         SET FLAG2,NEXT/IE.INST.FAULT    ; NO MORE PARAMETERS TO PUSH.
;38700
;38701 .REGION/IRD1.R1L,IRD1.R1H
;38702 =000
;38703 IE.OPCOD.DEC:
;38704         ;-------------------------------;
;38705         D_M[SCBB]+ZLIT0[10],            ; LOAD THE MACRO VECTOR ADDRESS.
U 03B0, 0D50,EC11,2030,8047,00F6,4      ;38706         SET FLAG2,NEXT/IE.INST.FAULT    ; NO MORE PARAMETERS TO PUSH.
;38707 =
;38708 .REGION/IANDE.R1L,IANDE.R1H/IANDE.R2L,IANDE.R2H/IANDE.R3L,IANDE.R3H
;38709 IE.OPCOD.CUST:
;38710         ;-------------------------------;
;38711         D_M[SCBB]+ZLIT0[14],            ; LOAD THE MACRO VECTOR ADDRESS.
U 0FA9, 0550,EC11,2030,A047,00F6,4      ;38712         SET FLAG2,NEXT/IE.INST.FAULT    ; NO MORE PARAMETERS TO PUSH.
```

```
                                ;38713 2C:     ;**UTRAP ADDR**                  ; FPA RESV OPER IS DETECTED BY A UTRAP
                                ;38714 IE.FPA.FAULT:
                                ;38715         ;-------------------------------;
                                ;38716         D_M[SCBB]+ZLIT0[18],             ; LOAD THE MACRO VECTOR ADDRESS.
U 002C, 0150,EC11,2030,C047,0001,2 ;38717      SET FLAG2,NEXT/IE.FPTCR          ; GO DECODE TRAPS VS FAULTS
                                ;38718
                                ;38719 IE.OPER.ABORT:
                                ;38720         ;-------------------------------;
                                ;38721         VA_M[SCBB]+ZLIT0[18],            ; LOAD THE MACRO VECTOR ADDRESS.
U 0FAA, 0D50,EC11,0030,C4A7,00EC,C ;38722      SET FLAG2,NEXT/IE.ABORT          ; NO MORE PARAMETERS TO PUSH.
                                ;38723
                                ;38724 0FF8:   ;*********FORCE ADDRESS*********;
                                ;38725 IE.OPER.FAULT:
                                ;38726 IE.FPA.FAULT.TRAP:
                                ;38727         ;111000---------------------------; RESERVED OPERAND FROM FPA.
                                ;38728         D_M[SCBB]+ZLIT0[18],             ; LOAD THE MACRO VECTOR ADDRESS.
                                ;38729         FPD?,                            ; ROLL BACK OR PACK UP?
U 0FF8, 0950,EC11,23F0,C047,00F6,4 ;38730      SET FLAG2,NEXT/IE.INST.FAULT     ; NO MORE PARAMETERS TO PUSH.
                                ;38731
                                ;38732 0FF9:   ;111001---------------------------; INTEGER OVERFLOW FROM FPA.
                                ;38733         M[ERRCOD]_CONX(1),               ; SET ERROR CODE
U 0FF9, 0486,B737,0000,0047,00F8,4 ;38734      NEXT/IE.ARITH.TRP                ; TAKE A TRAP.
                                ;38735
                                ;38736 0FFB:   ;*********FORCE ADDRESS*********;
                                ;38737 IE.FLOV.FAULT:
                                ;38738         ;111011---------------------------;
                                ;38739         M[ERRCOD]_ZLIT0[8],CLEAR FLAG2,  ; SET ERROR CODE & INDICATE MORE PARAMS
U 0FFB, 0516,BC37,0030,4047,00FA,B ;38740      NEXT/IE.FLT.FAULT                ;
                                ;38741
                                ;38742 0FFC:   ;*********FORCE ADDRESS*********;
                                ;38743 IE.FLDBZ.FAULT:
                                ;38744         ;111100---------------------------;
                                ;38745         M[ERRCOD]_ZLIT0[9],CLEAR FLAG2,  ; SET ERROR CODE & INDICATE MORE PARAMS
U 0FFC, 0116,BC37,0030,4847,00FA,B ;38746      NEXT/IE.FLT.FAULT                ;
                                ;38747
                                ;38748 0FFD:   ;*********FORCE ADDRESS*********;
                                ;38749 IE.FLUN.FAULT:
                                ;38750         ;111101---------------------------;
                                ;38751         M[ERRCOD]_ZLIT0[0A],CLEAR FLAG2, ; SET ERROR CODE & INDICATE MORE PARAMS
U 0FFD, 0116,BC37,0030,5047,00FA,B ;38752      NEXT/IE.FLT.FAULT                ;
                                ;38753
                                ;38754 IE.FLT.FAULT:
                                ;38755         ;-------------------------------;
                                ;38756         D_M[SCBB]+ZLIT0[34],             ; LOAD THE MACRO VECTOR ADDRESS.
U 0FAB, 0980,EC11,23F1,A047,00F6,4 ;38757      FPD?,NEXT/IE.INST.FAULT          ;
```

```
                                        ;38758 0F64:   ;*********FORCE ADDRESS*********;
                                        ;38759 IE.INST.FAULT:
                                        ;38760         ;00-----------------------------; BEGIN RBS ROLL BACK.
                                        ;38761         R[TEMP0] RBSP,                   ; SAVE RBSP TO LOAD INTO THE STEPC.
                                        ;38762         RBSP.EQ.0?,                      ; CHECK FOR RBSP = 0.
U 0F64, 0885,E036,4BB0,0047,04F5,A      ;38763         PUSH,NEXT/IE.RBS.RBACK           ; ROLL BACK THE GPR SIDE EFFECTS.
                                        ;38764
                                        ;38765 0F65:   ;01-----------------------------;
                                        ;38766         M[FPDOFFSET] MB+ZLIT8[7],        ; SET CODE FOR RETURN FROM FPD.PACK
U 0F65, 0D86,CD91,0230,3847,08FC,0 377* ;38767         WB<5-0>?,NEXT/IE.FPD.PACK        ; FPD WAS SET, PACK IT UP.
                                        ;38768
                                        ;38769 0F66:   ;10-----------------------------; RBS.RBACK DOES RETURN + 2.
                                        ;38770         VA_D+R[ZERO],                    ; LOAD MACRO VECTOR ADDRESS.
                                        ;38771         CLEAR FLAG3,                     ; PUSH PCBACK - 2.
U 0F66, 0018,0021,003D,84A7,04EC,D      ;38772         PUSH,NEXT/IE.FAULT               ; GO INITIATE THE EXCEPTION.
                                        ;38773
                                        ;38774 0F67:   ;11-----------------------------;
                                        ;38775         VA_R[SP] RB-CONX(4),             ; PREPARE TO PUSH PARAMETER
U 0F67, 0C84,073C,0027,84A7,00FA,7      ;38776         NEXT/IE.ARITH.PSH.CODE           ;
```

```
                                ;38777 .TOC "  Interrupts and Exceptions       : INTERVAL TIMER SERVICE"
                                ;38778
                                ;38779 ;*************************************************************************
                                ;38780 ;       Entry points    IE.ICR.DOSERV   hardware entry point
                                ;38781 ;                       IE.ICR.SERV     micro code entry point
                                ;38782 ;
                                ;38783 ;       Input           MM.TEMP0        TCSR.IICR
                                ;38784 ;
                                ;38785 ;       Output          MM.TEMP0        TCSR.IICR
                                ;38786 ;
                                ;38787 ;       Resources       SPNICR.SPCIR
                                ;38788 ;                       TCSR.IICR
                                ;38789 ;
                                ;38790 ;       This routine does micro service of the interval timer.
                                ;38791 ;       It has two functions, transfer the SPNICR into the SPICR and
                                ;38792 ;       increment the SPICR.
                                ;38793 ;
                                ;38794 ;       SPNICR is the upper 16 bits of the NICR, and
                                ;38795 ;       SPICR is the upper 16 bits of the ICR.
                                ;38796 ;       These are both stored in one 32 bit register called SPNICR.SPICR.
                                ;38797 ;       SPNICR.SPICR actually contains SPICR+1 in the lower 16 bits and
                                ;38798 ;       SPNICR in the upper 16 bits.  This is done so that when SPICR is
                                ;38799 ;       incremented to a state of all ones ( time to set VP ) the stored value
                                ;38800 ;       will be zero making testing easy (MFPR must take this into account).
                                ;38801 ;*************************************************************************
                                ;38802
                                ;38803 14:                                      ; FORCE ADDRESS FOR HARDWARE BRANCH.
                                ;38804 IE.ICR.DOSERV:
                                ;38805         ;-------------------------------;
                                ;38806         PC_M[PCBACK]-ZLIT0[2],          ; BACKUP PC BECAUSE IRD1 WILL BE
U 0014, 0D81,9C10,0030,1487,00F8,F ;38807     NEXT/IE.ICR.DOSERV2             ; DONE AT COMPLETION ON TIMER SERVICE.
                                ;38808
                                ;38809 0F8E:   ;*********FORCE ADDRESS*********;
                                ;38810         ;0-------------------------------; CATCH RETURN-1 FROM DOSERV DETECTED
U 0F8E, 0080,0036,4130,0047,003F,9 ;38811     IRD1                            ; TIMER SERVICE.
                                ;38812
                                ;38813 0F8F:   ;*********FORCE ADDRESS*********;
                                ;38814 IE.ICR.DOSERV2:
                                ;38815         ;1-------------------------------;
                                ;38816         M[MM.TEMP0]_TCSR.IICR,          ; BEGIN TIMER SERVICE DETECTED
U 0F8F, 0886,D036,4030,01E7,04FE,0 ;38817     PUSH,NEXT/IE.ICR.SERV           ; DURING DOSERV OF IRD1.
                                ;38818
                                ;38819 0FE0:   ;*********FORCE ADDRESS*********;
                                ;38820 IE.ICR.SERV:
                                ;38821         ;-------------------------------;
                                ;38822         WB_.NOT.(M[MM.TEMP0].RR.16),    ;
U 0FE0, 0080,D3B6,0210,0047,08FB,3 337* ;38823 WB<5-0>?                        ; BRANCH ON TCSR<SR,TR>
```

```
                                        ;38824 0FB3:   ;*********FORCE ADDRESS*********;
                                        ;38825 IE.ICR.SERV2:
                                        ;38826         ;110011-------------------------;
U 0FB3, 0482,02B7,001B,8047,00FA,C      ;38827         SPICR_SPNICR,NEXT/IE.ICR.INC    ;
                                        ;38828
                                        ;38829 0FB7:   ;*********FORCE ADDRESS*********;
                                        ;38830 IE.ICR.SR:
                                        ;38831         ;110111-------------------------; TCSR<SR> IS SET, INCREMENT SPICR.
                                        ;38832         R[SPNICR.SPICR].SIZ_RB+1,       ; INCREMENT SPICR, DON'T CHANGE SPNICR.
                                        ;38833         SIZE[WORD],SIGND CMP?,          ; IF RESULT WORD IS ZERO, THE CLOCK
U 0FB7, 0882,0E7C,0B5B,8047,08F4,2 407* ;38834         NEXT/IE.ICR.VP                  ; IS ABOUT TO OVERFLOW, SET TCSR<VP>.
                                        ;38835
                                        ;38836 0FBB:   ;111011-------------------------; TCSR<TR> IS SET, DO A TRANSFER.
U 0FBB, 0482,02B7,001B,8047,00FB,7      ;38837         SPICR_SPNICR,NEXT/IE.ICR.SR     ; TRANSFER SCRATCH PAD PORTION OF ICR.
                                        ;38838
                                        ;38839 0FBF:   ;111111-------------------------;
U 0FBF, 0080,0036,40B0,0047,03FF,F      ;38840         RETURN [-1]                     ; RETURN - 1
                                        ;38841
                                        ;38842 IE.ICR.INC:
                                        ;38843         ;-------------------------------;
                                        ;38844         R[SPNICR.SPICR].SIZ_RB+1,       ; INCREMENT SPICR, DON'T CHANGE SPNICR.
U 0FAC, 0082,0E7C,001B,8047,00FB,7      ;38845         SIZE[WORD],NEXT/IE.ICR.SR       ;
                                        ;38846
                                        ;38847 =10
                                        ;38848 IE.ICR.VP:
                                        ;38849         ;10-------------------------------; ICR IS NOT ABOUT TO OVERFLOW,
                                        ;38850         M[MM.TEMP0]_MB.ANDNOT.OLIT16[80], ; DON'T WRITE A 1 TO TCSR<IR>,
                                        ;38851         WCTRL/TCSR_WB,                  ; WRITE TCSR CLEARING SR AND TR.
U 0F42, 0D86,DF13,80B4,0167,03FF,F      ;38852         RETURN [-1]                     ; RETURN-1
                                        ;38853
                                        ;38854         ;11-------------------------------; ICR IS ABOUT TO OVERFLOW,
                                        ;38855         M[MM.TEMP0]_MB.OR.ZLIT16[2],    ; SET TCSR<VP>
U 0F43, 0986,DD12,4030,1047,00F4,2      ;38856         NEXT/IE.ICR.VP                  ;
```

```
                                ;38857 .TOC "  Interrupts and Exceptions       : SERVICE INTERRUPT PENDING OR TIMER"
                                ;38858
                                ;38859 ;*****************************************************************************
                                ;38860 ;       Entry points    IE.SERV.IP.TS
                                ;38861 ;                       IE.SERV.IP.TS2
                                ;38862 ;
                                ;38863 ;       Input           FPDOFFSET       (not if timer service only)
                                ;38864 ;
                                ;38865 ;       Resources       MM.TEMP0        Save TCSR.IICR when servicing timer
                                ;38866 ;                       TEMP0           Save RBSP when rolling back
                                ;38867 ;
                                ;38868 ;       Subroutines     IE.RBS.RBACK    Roll back register side effects
                                ;38869 ;                       IE.FPD.PACK     Pack up FPD instruction
                                ;38870 ;                       IE.INT.WIDE.BRANCH      Service the interrupt
                                ;38871 ;
                                ;38872 ;       If the microcode execution flows detect timer service or interrupt
                                ;38873 ;       pending it can pass control here and all packing/rollback will be
                                ;38874 ;       taken care of and the interrupt initiated. If timer service is
                                ;38875 ;       needed it will be taken before the interrupt and if only timer service
                                ;38876 ;       is needed control will return to the execution flows with a
                                ;38877 ;       RETURN-1.
                                ;38878 ;*****************************************************************************
                                ;38879
                                ;38880 0F6D:   ;*********FORCE ADDRESS*********;
                                ;38881 IE.SERV.IP.TS:
                                ;38882         ;01-----------------------------; NOT CONSOLE MODE, PROCESS INTERRUPT
                                ;38883         INTPEND OR TIMER?,              ;
U 0F6D, 0C80,0036,4AF0,0047,00F7,0 ;38884      NEXT/IE.SERV.IP.TS2             ;
                                ;38885
                                ;38886 0F6F:   ;11-----------------------------; CONSOLE MODE, SIMPLY RETURN
                                ;38887         COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG CLEARING TB
U 0F6F, 0880,0036,40B0,0057,03FF,F ;38888      RETURN [-1]                     ;
                                ;38889
                                ;38890 0F70:   ;*********FORCE ADDRESS*********;
                                ;38891 IE.SERV.IP.TS2:
                                ;38892         ;00-----------------------------; RETURN-1 FROM IE.ECR.SERV,
                                ;38893         CLEAR FP TRAPS,                 ; CLEAR POSSIBLE ERRONEOUS FPA TRAPS
U 0F70, 0C80,0036,43F0,0067,002A,B ;38894      FPD?,NEXT/IE.SERV.FIX.VA        ; CHECK TO SEE IF THE VA NEEDS FIXING
                                ;38895
                                ;38896 0F71:   ;01-----------------------------; TIMER SERVICE AND INTERRUPT, FIRST DO
                                ;38897         M[MM.TEMP0]_TCSR.IICR,          ; TIMER SERVICE, THEN RETURN-1 AND
U 0F71, 0886,D036,4030,01E7,04FE,0 ;38898      PUSH,NEXT/IE.ICR.SERV           ; SEVICE THE INTERRUPT.
                                ;38899
                                ;38900 0F72:   ;10-----------------------------; NO INTERRUPT OR TIMER SERVICE,
                                ;38901         COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG CLEARING TB
U 0F72, 0880,0036,40B0,0057,03FF,F ;38902      RETURN [-1]                     ;
                                ;38903
                                ;38904 0F73:   ;11-----------------------------; TIMER SERVICE ONLY,
                                ;38905         M[MM.TEMP0]_TCSR.IICR,          ; TIMER SERVICE WILL RETURN-1 TO
U 0F73, 0086,D036,4030,01E7,00FE,0 ;38906      NEXT/IE.ICR.SERV                ; CALLING ROUTINE
```

```
                                        ;38907 02AB:
                                        ;38908 IE.SERV.FIX.VA:
                                        ;38909          ;1********FORCE ADDRESS*********; IF FAULT FROM READ SECOND
U 02AB, 0080,0036,46B0,0047,0000,C      ;38910          STACK FLAG?                      ; CHECK IF VA NEEDS CORRECTING
                                        ;38911
                                        ;38912 000C:    ;0********FORCE ADDRESS*********;
U 000C, 0C80,0036,43F0,0047,00F8,0      ;38913          FPD?,NEXT/IE.SERV.IP             ; ROLL BACK RBS OR FPD THE INSTRUCTION.
                                        ;38914
                                        ;38915 000D:    ;1********FORCE ADDRESS*********;
                                        ;38916          VA_M[VA]-ZLIT0[4],               ; FIX VA FROM UNALIGNED READ
                                        ;38917          CLEAR STACK FLAG,                ; CLEAR TO AVOID ERRONEOUS KSNV
U 000D, 0529,BC10,03F0,24A7,00F8,0      ;38918          FPD?,NEXT/IE.SERV.IP             ; ROLL BACK RBS OR FPD THE INSTRUCTION.
                                        ;38919 =00
                                        ;38920 IE.SERV.IP:
                                        ;38921          ;00-----------------------------; FPD IS CLEAR.
                                        ;38922          R[TEMP0] RBSP,                   ; SAVE RBSP TO LOAD INTO THE STEPC.
                                        ;38923          RBSP.EQ.0?,CLEAR FLAG3,          ; CHECK FOR RBSP = 0.
U 0F80, 001D,E036,48B0,0047,04F5,A      ;38924          PUSH,NEXT/IE.RBS.RBACK           ; ROLL BACK THE GPR SIDE EFFECTS.
                                        ;38925
                                        ;38926          ;01-----------------------------; FPD IS SET.
                                        ;38927          M[FPDOFFSET] MB+ZLIT8[4],        ; SET CODE FOR RETURN FROM FPD.PACK
U 0F81, 0D86,CD91,0230,2047,08FC,0 377* ;38928          WB<5-0>?,NEXT/IE.FPD.PACK        ; PACK IT UP.
                                        ;38929
                                        ;38930          ;10-----------------------------; RETURN FROM RBS.RBACK
                                        ;38931          MICRO VECTOR?,                   ; BRANCH TO INTERRUPT ROUTINES.
                                        ;38932          CLEAR TP,                        ; CLEAR PSL<TP> ON INTERRUPTS.
U 0F82, 0C90,0036,47B0,0047,0803,8 479* ;38933          NEXT/IE.INT.WIDE.BRANCH          ;
                                        ;38934 =
```

```
                                        ;38935 .TOC "  Interrupts and Exceptions       : IE.RBS.RBACK"
                                        ;38936
                                        ;38937 ;*****************************************************************************
                                        ;38938 ;       Entry points    IE.RBS.RBACK
                                        ;38939 ;
                                        ;38940 ;       Input           TEMP0           RBSP
                                        ;38941 ;
                                        ;38942 ;       Resources       STEPC
                                        ;38943 ;                       RNUM
                                        ;38944 ;
                                        ;38945 ;       IE.RBS.RBACK is a routine to roll back the gpr side effects recorded
                                        ;38946 ;       in the register back up stack (RBS), in order to restart an instruction.
                                        ;38947 ;*****************************************************************************
                                        ;38948
                                        ;38949 =0
                                        ;38950 IE.RBS.RBACK:
                                        ;38951         ;0-------------------------------;
                                        ;38952         STEPC_M[TEMP0]<3-0>.,RBSP_0,    ; LOAD RBSP INTO THE STEPC.
U 0F5A, 0080,03F7,003D,C107,00F7,4      ;38953         NEXT/IE.RBS.LOOP                ;
                                        ;38954
                                        ;38955         ;1-------------------------------; THE RBSP IS ZERO, THERE ARE NO
U 0F5B, 0880,0036,40B0,0047,0000,2      ;38956         RETURN [+2]                     ; REGISTERS TO BACK UP, JUST RETURN.
                                        ;38957
                                        ;38958 =0
                                        ;38959 IE.RBS.LOOP:
                                        ;38960         ;0-------------------------------;
                                        ;38961         RNUM_POPRBS MTEMP0_POPSIZ,      ;
U 0F74, 0887,C036,4BB0,0047,00F0,4      ;38962         RBS + OR - ?,NEXT/IE.RBS.UNDO   ; TEST FOR INCREMENT OR DECREMENT.
                                        ;38963
                                        ;38964         ;1-------------------------------; THE STEP COUNTER IS ZERO, THE
U 0F75, 0880,0036,40B0,0047,0000,2      ;38965         RETURN [+2]                     ; BACK UP IS COMPLETE.
                                        ;38966
                                        ;38967 =0*
                                        ;38968 IE.RBS.UNDO:
                                        ;38969         ;0*------------------------------; THE GPR WAS DECREMENTED,
                                        ;38970         R[GPR.R]_M[TEMP0]<3-0>+RB,      ; ADD THE SAVED SIZE BACK IN.
U 0F04, 0084,03FD,033C,C047,00F7,4      ;38971         DBZ STEPC?,NEXT/IE.RBS.LOOP     ; CONTINUE POPPING RBS UNTIL STEPC=0.
                                        ;38972
                                        ;38973         ;1*------------------------------; THE GPR WAS INCREMENTED,
                                        ;38974         R[GPR.R]_(RB-M[TEMP0]<3-0>),    ; SUBTRACT THE SAVED SIZE OUT.
U 0F06, 0484,03FC,033C,C047,00F7,4      ;38975         DBZ STEPC?,NEXT/IE.RBS.LOOP     ; CONTINUE POPPING RBS UNTIL STEPC=0.
```

```
                                ;38976 .TOC "  Interrupts and Exceptions       : IE.FPD.PACK"
                                ;38977
                                ;38978 ;*****************************************************************************
                                ;38979 ;       Entry Points    IE.FPD.PACK
                                ;38980 ;
                                ;38981 ;       This is a table of the first instructions of FPD Packing routines.
                                ;38982 ;       It is used by driving FPDOFFSET onto the WBUS and branching on
                                ;38983 ;       WBUS<5:0>.
                                ;38984 ;
                                ;38985 ;       This table is branched to from 4 places, IE.SERV.IP.TS (initiate an
                                ;38986 ;       interrupt), IE.ACV (initiate ACV/TNV), IE.OPER.FAULT (initiate
                                ;38987 ;       reserved operand fault), and IE.MACH.CHK.RBACK (machine check fault).
                                ;38988 ;       THESE ROUTINES DO NOT DO A PUSH TO GET HERE.
                                ;38989 ;       This is to allow some pack routines to regain control by doing
                                ;38990 ;       a RETURN, the normal thing is to jump to IE.PACK.DONE which
                                ;38991 ;       will return control to the proper initiation routine using
                                ;38992 ;       the code saved in FPDOFFSET<10:8> (see IE.PACK.DONE).
                                ;38993 ;*****************************************************************************
                                ;38994
                                ;38995 .SEQUENTIAL
                                ;38996 OFC0:
                                ;38997 IE.FPD.PACK:
                                ;38998         ;000000-------------------------;
U OFC0, 0880,0036,40B0,0047,0000,4 ;38999       RETURN [+4]                     ; EDITPC (RETURN + 4)
                                ;39000
                                ;39001         ;000001-------------------------; POLYF PACK ROUTINE
                                ;39002         R[R2].SIZ_RB+1,SIZE[3YTE],      ;
U OFC1, 0482,0E7C,0004,8047,0070,2 ;39003       NEXT/FP.POLYF.FPD.SAV.10        ;
                                ;39004
                                ;39005         ;000010-------------------------; POLYD PACK ROUTINE
                                ;39006         R[R2].SIZ_RB+1,SIZE[BYTE],      ;
                                ;39007         READ(FLAG1)?,                   ; WAS THIS SPECIAL CASE
U OFC2, 0482,0E7C,05C4,8047,0061,8 ;39008       NEXT/FP.POLYD.FPD.SAV.10        ;
                                ;39009
                                ;39010         ;000011-------------------------; CRC PACK ROUTINE
U OFC3, 0C85,A592,4034,C047,0071,7 ;39011       R[R3]_M[PC],NEXT/CR.CRC.SAVE    ; SAVE THE STREAM ADDRESS.
                                ;39012
                                ;39013         ;000100-------------------------; BUT XB TBMISS WITH FPD=1, DO NOT PACK.
                                ;39014         CLEAR FLAG3,                    ; PUSH PCBACK-2
                                ;39015         WB_M[FPDOFFSET].RR.8,           ; SPLIT FLOW BACK INTO INTERRUPT OR
U OFC4, 0418,C3B7,0200,0047,08F5,0 337* ;39016  WB<5-0>?,NEXT/IE.PACK.OR.RBACK  ; EXCEPTION FLOW THAT CALLED IE.FPD.PACK
                                ;39017
                                ;39018 CS.SC.SPAN.PACK:
                                ;39019         ;000101-------------------------;SCAN SPAN BONNIE
U OFC5, 0480,0036,4030,0047,00B9,2 ;39020       NEXT/CS.SC.SPAN.PACK.BEGI       ;GO TO PACK ROUTINE FOR SCAN SPAN
                                ;39021
                                ;39022 CS.LOCC.SKPC.PACK:
                                ;39023         ;000110-------------------------;LOCC SKPC BONNIE
                                ;39024         M[TEMP5]_MB.RR.16,              ;MTMP5_?'FILL'?'?
U OFC6, 0C86,53B7,0010,0047,00BD,E ;39025       NEXT/CS.LOCC.SKPC.PACK.BE       ;
```

```
                                        ;39026 CS.MATCHC.PACK:
                                        ;39027         ;000111-------------------------;
                                        ;39028         R[R2]_M[TEMP8].AND.RB,          ;R2_00 IN BITS 31-30 IN R2
U 0FC7, 0484,8002,0034,8047,00FE,2      ;39029         NEXT/IE.PACK.DONE               ;NO PACKING JUST RET-1 TO CHARLIE
                                        ;39030
                                        ;39031 CS.CMP.PACK:
                                        ;39032         ;001000-------------------------;CMPC3 CMPC5 BONNIE
                                        ;39033         M[TEMP2]_FLAGS,                 ;TEMP2 GETS FLAGS
U 0FC8, 0886,2036,4030,0387,00BC,A      ;39034         NEXT/CS.CMP.PACK.BEGIN          ;
                                        ;39035
                                        ;39036 CS.MOV.PACK:
                                        ;39037         ;001001-------------------------;MOVC3 MOVC5 MOVTC MOVTUC BONNIE
                                        ;39038         M[TEMP5]_MB.AND.ZLIT16[0FF],    ;MTMP5_0'FILL'0'0
U 0FC9, 0986,5D12,0037,F847,00BA,6      ;39039         NEXT/CS.MOV.PACK.BEGIN          ;
                                        ;39040
                                        ;39041 DS.CVT.PACK:
                                        ;39042         ;001010-------------------------;CVTPS, CVTSP, CVTTP, CVTPT
                                        ;39043         MTEMP0_R[TEMP8]-PCBACK,         ; COMPUTE PC OFFSET
U 0FCA, 0087,9003,0032,0047,0137,0      ;39044         NEXT/DS.CVTXX.PACK              ;
                                        ;39045
                                        ;39046         ;001011-------------------------;MOVP, CMPP, ADDP, SUBP, MULP, ASHP,DIVP
                                        ;39047         CLEAR FLAG3,                    ; PUSH PCBACK-2
                                        ;39048         WB_M[FPDOFFSET].RR.8,           ; SPLIT FLOW BACK INTO INTERRUPT OR
U 0FCB, 0418,C3B7,0200,0047,08F5,0 337* ;39049         WB<5-0>?,NEXT/IE.PACK.OR.RBACK  ; EXCEPTION FLOW THAT CALLED IE.FPD.PACK
                                        ;39050
                                        ;39051 FI.POLYF.FPD.SAVE:
                                        ;39052         ;001100-------------------------;
                                        ;39053         R[R2].SIZ_RB+1,                 ; INCREMENT ITERATION COUNT
                                        ;39054         SIZE[BYTE],                     ;
                                        ;39055         FPA(FLAG0)?,                    ; FAULT DETECTED BY FPA?
U 0FCC, 0082,0E7C,0404,8047,0024,C      ;39056         NEXT/FI.POLYF.FPD.10            ; CONTINUE ON
                                        ;39057
                                        ;39058 FI.POLYD.FPD.SAVE:
                                        ;39059         ;001101-------------------------;
                                        ;39060         R[R2].SIZ_RB+1,                 ; INCREMENT ITERATION COUNT
                                        ;39061         SIZE[BYTE],                     ; SECOND HALF READ OR
                                        ;39062         READ(FLAG1) FPA(FLAG0)?,        ; FAULT DETECTED BY FPA?
U 0FCD, 0482,0E7C,0504,8047,001F,4      ;39063         NEXT/FI.POLYD.FPD.10            ; CONTINUE ON
                                        ;39064
                                        ;39065         ;001110-------------------------;
                                        ;39066         R[R2]_M[TEMP9],                 ;
U 0FCE, 0C84,9592,4034,8047,0139,5      ;39067         NEXT/DS.DS.PCK.GPR2.2-3         ;
                                        ;39068
                                        ;39069         ;001111-------------------------;
                                        ;39070         R[R2]_M[TEMP9],                 ;
U 0FCF, 0484,9592,4034,8047,0139,7      ;39071         NEXT/DS.DS.PCK.GPR2.3CON        ;
                                        ;39072
                                        ;39073         ;010000-------------------------;
                                        ;39074         M[TEMP3]_R[TEMP8].RR.24,        ;
U 0FD0, 0C86,3287,0022,0047,013A,3      ;39075         NEXT/DS.DS.PCK.MULP             ;
```

```
                                ;39076 DS.DS.PCK.GPR0:
                                ;39077          ;010001-------------------------;
                                ;39078          R[R0]_M[PC],                    ;
U 0FD1, 0485,A592,4034,0047,0139,1 ;39079       NEXT/DS.DS.PCK.GPR0.CON         ;
                                ;39080
                                ;39081          ;010010-------------------------;
                                ;39082          R[R0]_RB-M[PCBACK],             ;
U 0FD2, 0485,9003,0034,0047,0139,E ;39083       NEXT/DS.DS.PCK.CMPP4            ;
                                ;39084
                                ;39085          ;010011-------------------------;
                                ;39086          VA_M[TEMP4],SET MM.NOINT,       ;
U 0FD3, 0460,4002,403D,84A7,0129,7 ;39087       NEXT/QU.RESV.OPER               ;
                                ;39088
                                ;39089          ;010100-------------------------;
                                ;39090          R[R0]_M[PC],IR<2-0>?,           ;
U 0FD4, 0885,A592,4674,0047,0103,5 ;39091       NEXT/DS.DS.PCK.CMPP34           ;
                                ;39092
                                ;39093
                                ;39094 FX.POLYG.FPD.SAVE:
                                ;39095          ;010101-------------------------; POLYG
                                ;39096          R[R2].SIZ_RB+1,                 ; INCREMENT ITERATION COUNT
                                ;39097          SIZE[BYTE],                     ;
                                ;39098          READ(FLAG1)?,                   ; IS IT SPECIAL READ
U 0FD5, 0482,0E7C,05C4,8047,0061,8 ;39099       NEXT/FX.POLYG.FPD.SAV.10        ; DEFINED IN FLOAT.MIC
                                ;39100
                                ;39101 FX.POLYH.FPD.SAVE:
                                ;39102          ;010110-------------------------; POLYH
                                ;39103          R[R4].SIZ_RB+1,                 ; INCREMENT ITERATION COUNT
                                ;39104          SIZE[BYTE],                     ;
                                ;39105          READ(FLAG1)?,                   ; IS IT SPECIAL READ
U 0FD6, 0482,0E7C,05C5,0047,0200,4 ;39106       NEXT/2004                       ;
                                ;39107 ;        NEXT/FX.POLYH.FPD.SAV.10        ; DEFINED IN WCS
                                ;39108
                                ;39109 FX.EMODH.FPD:
                                ;39110          ;010111-------------------------; EMODH
                                ;39111          WB_M[TEMP1]-1,                  ; WAS THE PSEUDO RBS USED
                                ;39112          WX.EQ.0?,                       ;
U 0FD7, 0C80,1E51,0A30,0047,0A00,8 393* ;39113  NEXT/2008                       ;
                                ;39114 ;        NEXT/FX.EMODH.FPD.RES           ; DEFINED IN WCS
                                ;39115
                                ;39116          ;011000-------------------------; CVTSP
U 0FD8, 0484,0E7C,0034,4047,00FC,A ;39117       R[R1]_RB+1,NEXT/DS.CVT.PACK     ;
                                ;39118 ,RANDOM
                                ;39119 0FEE:    ;101110***FORCE ADDRESS*********; CVTTP DESTINATION LENGTH = 0
U 0FEE, 0D86,BF37,0030,8047,0037,B ;39120       M[ERRCOD]_OLIT16[16.]           ; FIELDS LEFT TO AVOID ROM CHANGE
                                ;39121
                                ;39122 037B:    ;*********FORCE ADDRESS*********; WORD ADDED TO AVOID ROM CHANGE
U 037B, 0440,0036,4030,0047,00FC,A ;39123       SET FLAG0,NEXT/DS.CVT.PACK      ; SET FLAG TO INDICATE DEST LEN=0
```

```
                                ;39124 .TOC "  Interrupts and Exceptions       : IE.PACK.DONE"
                                ;39125
                                ;39126 ;*****************************************************************************
                                ;39127 ;       Entry Points    IE.PACK.DONE
                                ;39128 ;
                                ;39129 ;       Input           FPDOFFSET<10:8> Return code
                                ;39130 ;
                                ;39131 ;       All FPD pack routines should branch here when packing is complete.
                                ;39132 ;       FPDOFFSET<10:8> contains a code which determines what sort of interrupt
                                ;39133 ;       or exception this is for:
                                ;39134 ;
                                ;39135 ;                       2 - ACV/TNV
                                ;39136 ;                       4 - INTERRUPT
                                ;39137 ;                       5 - MACHINE CHECK FAULT/ABORT
                                ;39138 ;                       7 - RESERVED OPERAND OR FLOAT FAULT
                                ;39139 ;
                                ;39140 ;*****************************************************************************
                                ;39141
                                ;39142 0FE2:   ;*********FORCE ADDRESS*********;
                                ;39143 IE.PACK.DONE:
                                ;39144         ;-------------------------------;
                                ;39145         CLEAR FLAG3,                    ; PUSH PCBAKC-2
                                ;39146         WB_M[FPDOFFSET].RR.8,           ; SPLIT FLOW BACK INTO INTERRUPT OR
U 0FE2, 0418,C3B7,0200,0047,08F5,0 337*  ;39147         WB<5-0>?                        ; EXCEPTION FLOW THAT CALLED IE.FPD.PACK
                                ;39148
                                ;39149 0F50:   ;*********FORCE ADDRESS*********;
                                ;39150 IE.PACK.OR.RBACK:
                                ;39151         ;0000---------------------------; THIS IS LOGICALLY PART OF ACV/TNV
                                ;39152         R[TEMP0]_RBSP,                  ; SAVE RBSP TO LOAD INTO THE STEPC.
                                ;39153         RBSP.EQ.0?,                     ; CHECK FOR RBSP = 0.
U 0F50, 0885,E036,4BB0,0047,04F5,A       ;39154         PUSH,NEXT/IE.RBS.RBACK          ; ROLL BACK THE REGISTER SIDE EFFECTS.
                                ;39155
                                ;39156 0F51:   ;0001---------------------------; THIS IS LOGICALLY PART OF ACV/TNV
                                ;39157         M[FPDOFFSET]_MB+ZLIT8[2],       ; SET CODE FOR RETURN FROM FPD.PACK
U 0F51, 0186,CD91,0230,1047,08FC,0 377*  ;39158         WB<5-0>?,NEXT/IE.FPD.PACK       ; FPD WAS SET, PACK IT UP.
                                ;39159
                                ;39160 0F52:   ;0010---------------------------; ACV/TNV RETURN FROM FPD.PACK/RBS.RBACK
                                ;39161         CLEAR FLAG2,                    ; MORE PARAMETERS TO PUSH.
                                ;39162         M[TEMP6]_R[MM.TEMP3],           ; SAVE FAULTING ADDRESS.
U 0F52, 0416,65BE,4039,C047,04FA,D       ;39163         PUSH,NEXT/IE.PACK.FAULT         ; INITIATE THE EXCEPTION.
                                ;39164
                                ;39165 0F53:   ;0011---------------------------; THIS IS LOGICALLY PART OF ACV/TNV
                                ;39166         VA_R[SP]_RB-CONX(4),            ; PREPARE TO PUSH THE FAULTING ADDRESS.
U 0F53, 0484,073C,0027,84A7,00F4,8       ;39167         NEXT/IE.ACV.PUSH.PARAM          ; RETURN TO ACV/TNV FLOWS
                                ;39168
                                ;39169 0F54:   ;*********FORCE ADDRESS*********;
                                ;39170 IE.PACK.BRANCH:
                                ;39171         ;0100---------------------------; FPD FROM INTERRUPT
                                ;39172         MICRO VECTOR?,                  ; BRANCH TO INTERRUPT ROUTINES.
                                ;39173         CLEAR TP,                       ; CLEAR PSL<TP> ON INTERRUPTS.
U 0F54, 0C90,0036,47B0,0047,0803,8 479*  ;39174         NEXT/IE.INT.WIDE.BRANCH         ; RETURN TO INTERRUPT FLOWS
```

```
                                ;39175 0F55:   ;0101----------------------------; FPD FROM MACHINE CHECK.
U 0F55, 0C80,0036,4030,0047,00ED,2  ;39176          NEXT/IE.MACH.CHK.FAULT          ; RETURN TO MACHINE CHECK MICRO CODE.
                                ;39177
                                ;39178 0F57:   ;0111----------------------------; FPD FROM RESV OPER OR FLOAT FAULT
                                ;39179          VA_D+R[ZERO],                   ; LOAD MACRO VECTOR ADDRESS.
                                ;39180          CLEAR FLAG3,                    ; PUSH PCBACK - 2.
U 0F57, 0018,0021,003D,84A7,04EC,D  ;39181          PUSH,NEXT/IE.FAULT              ; GO INITIATE THE EXCEPTION.
                                ;39182
                                ;39183 0F58:   ;1000----------------------------;
                                ;39184          VA_R[SP]_RB-CONX(4),            ; PREPARE TO PUSH PARAMETER
U 0F58, 0C84,073C,0027,84A7,00FA,7  ;39185          NEXT/IE.ARITH.PSH.CODE          ;
                                ;39186
                                ;39187 IE.PACK.FAULT:
                                ;39188          ;-------------------------------;
                                ;39189          VA_D+R[ZERO],                   ; LOAD MACRO VECTOR ADDRESS.
                                ;39190          CLEAR FLAG3,                    ; PUSH PCBACK - 2.
U 0FAD, 0818,0021,003D,84A7,00EC,D  ;39191          NEXT/IE.FAULT                   ; GO INITIATE THE EXCEPTION.
```

```
                                ;39192 .TOC "  Interrupts and Exceptions       : FD OPCODE ENTRY"
                                ;39193 ;*************************************************************************
                                ;39194 ;
                                ;39195 ;       This code checks if WCS is present and enabled, if it is not an
                                ;39196 ;       opcode reserved to Digital fault is taken.  If WCS is present and
                                ;39197 ;       enabled, a wide branch is done into WCS on the second byte
                                ;39198 ;       of the opcode.
                                ;39199 ;
                                ;39200 ;*************************************************************************
                                ;39201
                                ;39202 03C0:   ;*********FORCE ADDRESS*********;
                                ;39203 FX.IE.FD.OPCOD:
                                ;39204         ;000-----------------------------;THIS WORD HAS NO EFFECT I REPEAT NONE
U 03C0, 0C80,0036,4030,05E7,007E,B ;39205         MDR_ZEXT(OSR)                   ; ******PATCH TO AVOID ROM CHANGES
                                ;39206
                                ;39207 07EB:   ;*********FORCE ADDRESS*********;
                                ;39208         ;-------------------------------;
                                ;39209         PC_M[PC]-ZLIT0[1],              ; MUST BUMP BACK FOR 2BYTE OPCODES
U 07EB, 0581,AC10,0030,0C87,007F,E ;39210         NEXT/FX.IE.FD.OPCOD2            ; GET TO A CONSTRAINED ADDRESS
                                ;39211
                                ;39212 03C4:   ;*********FORCE ADDRESS*********;
                                ;39213 FX.IE.FD.OPCOD4:
                                ;39214         ;-------------------------------;
                                ;39215         Q_M[MDR],                       ; MOVE TO WORKING MTEMP
                                ;39216         WCS DISABLED?,                  ; IS WCS PRESENT AND ENABLED
U 03C4, 0C81,2592,59F0,0047,003C,2 ;39217         NEXT/FX.IE.WCS.ENABLED          ;
                                ;39218
                                ;39219 03C2:   ;*********FORCE ADDRESS*********;
                                ;39220 FX.IE.WCS.ENABLED:
                                ;39221         ;0-----------------------------; YES
                                ;39222         M[TEMP10] Q,                    ; NOW IN STORAGE FOR USE
                                ;39223         BRA ON ADD?,                    ; DO WIDE BRANCH ON OPCODE
U 03C2, 0486,A03A,463D,8047,0201,0 ;39224         NEXT/2010                       ;
                                ;39225 ;       NEXT/FX.WCS.OPCODE              ; OFF TO WCS
                                ;39226
                                ;39227 03C3:   ;1-----------------------------; NO
U 03C3, 0C80,0036,4030,0047,003B,0 ;39228         NEXT/IE.OPCOD.DEC               ; NOT HERE
                                ;39229
                                ;39230 776:    ;*********FORCE ADDRESS*********;
                                ;39231 FX.IE.FD.RET:
                                ;39232         ;-------------------------------;
U 0776, 0880,0036,40B0,0047,0000,1 ;39233         RETURN [+1]                     ; TARGET FOR CNT0,CNT1
                                ;39234
                                ;39235 7FE:    ;*********FORCE ADDRESS*********;
                                ;39236 FX.IE.FD.OPCOD2:
                                ;39237         ;-------------------------------;
                                ;39238         LOD BRA?,                       ; GET THE TRUE OP CODE LOADED
U 07FE, 0880,0036,41F0,0047,007F,F ;39239         NEXT/FX.IE.FD.OPCOD3            ;
                                ;39240
                                ;39241 7FF:    ;*********FORCE ADDRESS*********;
                                ;39242 FX.IE.FD.OPCOD3:
                                ;39243         ;-------------------------------;
                                ;39244         MDR_ZEXT(OSR),                  ; GET THE TRUE OPCODE IN 2ND BYTE
U 07FF, 0C80,0036,4030,05E7,003C,4 ;39245         NEXT/FX.IE.FD.OPCOD4            ; CONSTRAINED ADDRESS
```

```
                                ;39246 .TOC "  Interrupts and Exceptions       : FD OPCODE ENTRY"
                                ;39247 ;*****************************************************************************
                                ;39248 ;
                                ;39249 ;
                                ;39250 ;                    FPD RESTART WITH OPCODE OF FDxx
                                ;39251 ;
                                ;39252 ;       This code checks to see if WCS is present and enabled.
                                ;39253 ;       If WCS is not present and enabled an opcode reserved to Digital
                                ;39254 ;       fault is taken.
                                ;39255 ;       If WCS is present and enabled control is passed to the unpacking
                                ;39256 ;       code in WCS.
                                ;39257 ;
                                ;39258 ;*****************************************************************************
                                ;39259
                                ;39260 03F0:    ;*********FORCE ADDRESS*********;
                                ;39261 FX.IE.FPD.RES:
                                ;39262          ;000-----------------------------;THIS WORD HAS NO EFFECT I REPEAT NONE
                                ;39263          MDR_ZEXT(OSR),                  ;********PATCH TO AVOID ROM CHANGES
U 03F0, 0C80,0036,49F0,05E7,007E,D ;39264       WCS_DISABLED?                   ;***********************************
                                ;39265
                                ;39266 07ED:    ;*********FORCE ADDRESS*********; CONSTRAINED TO NULLIFY BRANCH
                                ;39267          ;-------------------------------;
                                ;39268          PC_M[PC]-ZLIT0[1],              ; BACK UP PC FOR 2 BYTE OPCODES
U 07ED, 0D81,AC10,0030,0C87,007E,E ;39269       NEXT/FX.IE.FPD.RES2             ; GET TO A CONSTRAINED ADDRESS
                                ;39270
                                ;39271 0FF4:    ;*********FORCE ADDRESS*********;
                                ;39272 FX.IE.FPD.RES4:
                                ;39273          ;0-------------------------------; WCS PRESENT AND ENABLED
                                ;39274          WB_M[MDR]-ZLIT0[55],            ; IS IT POLYG FPD
                                ;39275          WX.NE.0?,                       ;
U 0FF4, 0D81,2C10,0A72,A847,0A00,2 393* ;39276  NEXT/2002                        ;
                                ;39277 ;        NEXT/FX.POLYGH.CHK              ; DEFINED IN WCS
                                ;39278
                                ;39279 0FF5:    ;1-------------------------------; NO WCS
U 0FF5, 0C80,0036,4030,0047,003B,0 ;39280       NEXT/IE.OPCOD.DEC               ; DO NOT RESET FPD
                                ;39281
                                ;39282
                                ;39283
                                ;39284 7EE:     ;*********FORCE ADDRESS*********;
                                ;39285 FX.IE.FPD.RES2:
                                ;39286          ;-------------------------------;
                                ;39287          LOD BRA?,                       ; GET THE TRUE OP CODE LOADED
U 07EE, 0080,0036,41F0,0047,007E,F ;39288       NEXT/FX.IE.FPD.RES3             ;
                                ;39289
                                ;39290 7EF:     ;*********FORCE ADDRESS*********;
                                ;39291 FX.IE.FPD.RES3:
                                ;39292          ;-------------------------------;
                                ;39293          MDR_ZEXT(OSR),                  ; GET THE 2ND BYTE OPCODE
                                ;39294          WCS_DISABLED?,                  ;
U 07EF, 0C80,0036,49F0,05E7,00FF,4 ;39295       NEXT/FX.IE.FPD.RES4             ; CONSTRAINED ADDRESS
```

```
                                ;39296 .TOC "  Interrupts and Exceptions       : REI"
                                ;39297
                                ;39298 ;*****************************************************************************
                                ;39299 ;       REI     Return from Exception or Interrupt
                                ;39300 ;
                                ;39301 ;       Input           (SP)            New PC
                                ;39302 ;                       (SP+4)          New PSL
                                ;39303 ;
                                ;39304 ;       Resources       TEMP5           IPL of highest pending soft interrupt
                                ;39305 ;                       MTMP9           Temporary to switch stacks
                                ;39306 ;                       RTMP8           Save new PC
                                ;39307 ;                       VA
                                ;39308 ;                       MDR
                                ;39309 ;                       PC
                                ;39310 ;                       SOFTIPR
                                ;39311 ;                       LONLIT
                                ;39312 ;                       D
                                ;39313 ;                       SP              Current Stack Pointer
                                ;39314 ;                       ISP             Interrupt Stack Pointer
                                ;39315 ;                       KSP             Kernal Stack Pointer
                                ;39316 ;                       ESP             Executive Stack Pointer
                                ;39317 ;                       SSP             Supervisor Stack Pointer
                                ;39318 ;                       USP             User Stack Pointer
                                ;39319 ;                       SISR            Software Interrupt Summary Register
                                ;39320 ;                       PSL             Program Status Longword
                                ;39321 ;
                                ;39322 ;       Subroutines     IE.OPER.FAULT
                                ;39323 ;                       IE.SOFT.IPL
                                ;39324 ;*****************************************************************************
                                ;39325
                                ;39326 .REGION/IRD1.R1L,IRD1.R1H
                                ;39327 =000
                                ;39328 IE.REI: ;--------------------------------; LOAD THE ADDRESS
U 03B8, 0480,05BE,4037,84A7,00FE,4      ;39329         VA_R[SP]                        ; OF THE TOP OF THE STACK
                                ;39330 =
                                ;39331 .REGION/IANDE.R1L,IANDE.R1H/IANDE.R2L,IANDE.R2H/IANDE.R3L,IANDE.R3H
                                ;39332 0FE4:   ;*********FORCE ADDRESS*********;
                                ;39333         ;--------------------------------;
                                ;39334         READ,SIZE[LONG],PUSH RBS+,      ;
U 0FE4, 0485,573D,0027,84B0,00FA,F      ;39335         VA_R[SP]_RB+CONX(4)             ; POP THE NEW PC OFF THE STACK
                                ;39336
                                ;39337         ;--------------------------------;
U 0FAF, 0485,2592,4022,0050,00FB,4      ;39338         READ,SIZE[LONG],R[TEMP8]_M[MDR] ; POP THE NEW PSL AND SAVE THE NEW PC.
                                ;39339
                                ;39340         ;--------------------------------;
U 0FB4, 0C85,573D,0027,8047,00FB,C      ;39341         R[SP]_RB+CONX(4),PUSH RBS+      ; FINISH THE POP.
                                ;39342
                                ;39343         ;--------------------------------;
U 0FBC, 0B80,067E,F807,F847,00FB,D      ;39344         LONLIT_[3020FF00]               ; LOAD MASK TO CHECK MBZ BITS IN NEW PSL
```

```
                                ;39345          ;--------------------------------;
                                ;39346          WB_M[MDR].AND.R[LONLIT],        ; CHECK THAT ALL MBZ BITS IN THE NEW
U 0FBD, 0C81,2002,0A3D,4047,00F7,A ;39347       WX.EQ.0?                        ; PSL ARE ZERO.
                                ;39348
                                ;39349 =0       ;0------------------------------;
                                ;39350          CLEAR FLAG3,                    ; PUSH PCBACK-2.
U 0F7A, 0418,0036,4030,0047,00FF,8 ;39351       NEXT/IE.OPER.FAULT              ; TAKE A RESERVED OPERAND FAULT.
                                ;39352
                                ;39353          ;1------------------------------; IF IN COMPATABILITY MODE CHECK
                                ;39354          WB_M[MDR],WB<31-30>?,           ; SOME MORE MBZ BITS.
U 0F7B, 0881,2592,46F0,0047,00F8,9 ;39355       NEXT/IE.REI.20
                                ;39356 =00
                                ;39357 IE.REI.10:
                                ;39358          ;00-----------------------------;
                                ;39359          CLEAR FLAG3,                    ; PUSH PCBACK-2.
U 0F88, 0418,0036,4030,0047,00FF,8 ;39360       NEXT/IE.OPER.FAULT              ; TAKE A RESERVED OPERAND FAULT.
                                ;39361
                                ;39362 IE.REI.20:
                                ;39363          ;01-----------------------------;
                                ;39364          WB_M[MDR],REI_CHECK?,           ; DO AN REI CHECK ON THE NEW PSL.
U 0F89, 0881,2592,47B0,06E9,08F6,8 479* ;39365  NEXT/IE.REI.50
                                ;39366
                                ;39367 =11      ;11-----------------------------; LOAD MASK TO CHECK COMPATABILITY
                                ;39368          LONLIT_[0F0000E0],              ; MODE MBZ BITS.
U 0F8B, 0B40,0787,FFF8,F847,00FD,B ;39369       SET FLAG0                       ; REMEMBER COMP MODE FOR LATER
                                ;39370
                                ;39371          ;--------------------------------;
U 0FDB, 0081,2002,203D,4047,00FD,D ;39372       D_M[MDR].AND.R[LONLIT]          ; SAVE CURMOD AND MBZ BITS
                                ;39373
                                ;39374          ;--------------------------------;
                                ;39375          WB_D.XOR.ZLIT24[3],             ; CHECK MBZ BITS AND CHECK CURMOD=3
U 0FDD, 0980,0CB3,4A30,1847,00F8,8 ;39376       WX.EQ.0?,NEXT/IE.REI.10         ;
                                ;39377
                                ;39378 =01*00
                                ;39379 IE.REI.50:
                                ;39380          ;01*00--------------------------; REI CHECK OK,
                                ;39381          M[TEMP9]_R[SP],PSL<IS.CURM>?,   ; PREPARE TO SWITCH STACKS.
U 0F68, 0C86,95BE,47B7,8207,08F1,8 479* ;39382  NEXT/IE.REI.60                   ;
                                ;39383
                                ;39384          ;01*01--------------------------; REI CHECK OK BUT PENDING AST,
                                ;39385          M[SISR]_MB.OR.ZLIT0[4],         ; POST SOFTWARE INTERRUPT AT IPL=2
U 0F69, 0586,FC12,4030,2047,04F4,0 ;39386       PUSH,NEXT/IE.SOFT.IPL           ; RECOMPUTE HIGHEST SOFTWARE IPL.
                                ;39387
                                ;39388          ;01*10--------------------------; REI CHECK NOT OK,
                                ;39389          CLEAR FLAG3,                    ; PUSH PCBACK-2.
U 0F6A, 0418,0036,4030,0047,00FF,8 ;39390       NEXT/IE.OPER.FAULT              ; TAKE A RESERVED OPERAND FAULT.
                                ;39391
                                ;39392          ;01*11--------------------------; REI CHECK NOT OK,
                                ;39393          CLEAR FLAG3,                    ; PUSH PCBACK-2.
U 0F6B, 0418,0036,4030,0047,00FF,8 ;39394       NEXT/IE.OPER.FAULT              ; TAKE A RESERVED OPERAND FAULT.
                                ;39395
                                ;39396 =11*01   ;11*01--------------------------; LOAD SOFTWARE IPR REGISTER USED BY THE
                                ;39397          SOFTIPR_M[TEMP5].RR.16,         ; HARDWARE FOR ARBITRATION.
U 0F79, 0480,53B7,0010,0787,00F6,8 ;39398       NEXT/IE.REI.50                  ;
                                ;39399 =
```

```
                                ;39400 =1000
                                ;39401 IE.REI.60:
                                ;39402          ;1000----------------------------;
                                ;39403          R[KSP]_M[TEMP9],                 ; SAVE CURRENT STACK POINTER.
U 0F18, 0C84,9592,4BF8,0047,00F8,6 ;39404       PSL<TP>?,NEXT/IE.REI.70          ;
                                ;39405
                                ;39406          ;1001----------------------------;
                                ;39407          R[ESP]_M[TEMP9],                 ; SAVE CURRENT STACK POINTER.
U 0F19, 0884,9592,4BF8,4047,00F8,6 ;39408       PSL<TP>?,NEXT/IE.REI.70          ;
                                ;39409
                                ;39410          ;1010----------------------------;
                                ;39411          R[SSP]_M[TEMP9],                 ; SAVE CURRENT STACK POINTER.
U 0F1A, 0884,9592,4BF8,8047,00F8,6 ;39412       PSL<TP>?,NEXT/IE.REI.70          ;
                                ;39413
                                ;39414          ;1011----------------------------;
                                ;39415          R[USP]_M[TEMP9],                 ; SAVE CURRENT STACK POINTER.
U 0F1B, 0C84,9592,4BF8,C047,00F8,6 ;39416       PSL<TP>?,NEXT/IE.REI.70          ;
                                ;39417
                                ;39418          ;1100----------------------------;
                                ;39419          R[ISP]_M[TEMP9],                 ; SAVE CURRENT STACK POINTER.
U 0F1C, 0884,9592,4BF9,0047,00F8,6 ;39420       PSL<TP>?,NEXT/IE.REI.70          ;
                                ;39421
                                ;39422
                                ;39423 =
                                ;39424 =0
                                ;39425 IE.REI.70:
                                ;39426          ;0-----------------------------;
U 0F86, 0881,2002,403D,8007,00FD,E ;39427       PSL_M[MDR],NEXT/IE.REI.80        ; LOAD NEW PSL
                                ;39428
                                ;39429          ;1-----------------------------;
U 0F87, 0581,2C92,4032,0007,00FD,E ;39430       PSL_M[MDR].OR.ZLIT24[40]         ; SET TRACE PENDING IN NEW PSL.
                                ;39431
                                ;39432 IE.REI.80:
                                ;39433          ;-----------------------------;
U 0FDE, 0080,05BE,4032,0487,00FD,F ;39434       PC_R[TEMP8]                      ; LOAD NEW PC
                                ;39435
                                ;39436          ;-----------------------------;
U 0FDF, 0C80,0036,47B0,0207,08FE,8 479* ;39437  PSL<IS.CURM>?                     ; BRANCH TO GET NEW STACK POINTER
                                ;39438
                                ;39439 0FE8:    ;*********FORCE ADDRESS*********;
                                ;39440          ;1000----------------------------;
                                ;39441          M[TEMP9]_R[KSP],                 ; GET NEW STACK POINTER.
U 0FE8, 0086,95BE,4038,0047,0023,2 ;39442       NEXT/IE.REI.90                   ;
                                ;39443
                                ;39444 0FE9:    ;1001----------------------------;
                                ;39445          M[TEMP9]_R[ESP],                 ; GET NEW STACK POINTER.
U 0FE9, 0486,95BE,4038,4047,0023,2 ;39446       NEXT/IE.REI.90                   ;
                                ;39447
                                ;39448 0FEA:    ;1010----------------------------;
                                ;39449          M[TEMP9]_R[SSP],                 ; GET NEW STACK POINTER.
U 0FEA, 0486,95BE,4038,8047,0023,2 ;39450       NEXT/IE.REI.90                   ;
```

```
                                ;39451 0FEB:    ;1011-----------------------------;
                                ;39452          M[TEMP9]_R[USP],                  ; GET NEW STACK POINTER.
                                ;39453          FLAG0?,                           ; CHECK IF COMPATABILITY MODE.
U 0FEB, 0486,95BE,4438,C047,0023,2  ;39454      NEXT/IE.REI.90                    ;
                                ;39455
                                ;39456 0FEC:    ;1100-----------------------------;
U 0FEC, 0080,0036,4130,0047,003F,9  ;39457      IRD1                              ; NO NEED TO SWITCH STACKS.
                                ;39458
                                ;39459 0F8C:
                                ;39460 FREE.0F8C:
                                ;39461          ;0******LOCATION NOT USED*******;
                                ;39462          R[SP]_M[TEMP9],                   ;
U 0F8C, 0084,9592,4137,8047,003F,9  ;39463      IRD1                              ;
                                ;39464
                                ;39465 0F8D:
                                ;39466 FREE.0F8D:
                                ;39467          ;1******LOCATION NOT USED*******;
                                ;39468          SL_[16.],STEPC_14.,               ; WORD NOT USED
U 0F8D, 05B0,0F76,4030,8047,0423,2  ;39469      PUSH
                                ;39470
                                ;39471 232:
                                ;39472 IE.REI.90:
                                ;39473          ;0********FORCE ADDRESS*********;  - PART OF REI INSTR -
                                ;39474          R[SP]_M[TEMP9],                   ; LOAD NEW STACK POINTER AND
U 0232, 0084,9592,4137,8047,003F,9  ;39475      IRD1                              ; BEGIN EXECUTING MACRO CODE.
                                ;39476
                                ;39477 233:     ;1********FORCE ADDRESS*********;  - PART OF REI INSTR -
                                ;39478          WB_R[TEMP8],WB<0>?,PUSH,          ; COMPAT. MODE CLEAR GPR BITS <31:16>
U 0233, 0080,05BE,42B2,0047,057F,4  ;39479      NEXT/IE.PC.CHECK                  ; CHECK FOR ODD PC
                                ;39480
                                ;39481 17F4:
                                ;39482 IE.PC.CHECK:
                                ;39483          ;0******FORCE ADDRESS**********; PC NOT ODD
                                ;39484          SL_[16.],STEPC_14.,               ; LOAD SL FOR EXTRACT.
U 17F4, 05B0,0F76,4030,8047,00FE,3  ;39485      NEXT/IE.CLR.GPRS                  ;
                                ;39486
                                ;39487 17F5:
                                ;39488          ;1******FORCE ADDRESS**********; ODD PC
U 17F5, 0C80,0036,4030,0047,00FD,9  ;39489      NEXT/IE.CM.ODD                    ;
                                ;39490
                                ;39491 IE.CLR.GPRS:
                                ;39492          ;-------------------------------; LOAD TEMP0,RNUM AND PL WITH 0.
U 0FE3, 0C85,DBFE,003D,8047,00FE,5  ;39493      RTEMP0_RNUM_PL_0                  ;
                                ;39494
                                ;39495 IE.CLR.GPRS15:
                                ;39496          ;-------------------------------;
U 0FE5, 0C84,00B7,003C,C047,00FE,6  ;39497      R[GPR.R]_RB.XZ                    ; CLEAR BITS 31-16 (PL=0, SL=16).
                                ;39498
                                ;39499          ;-------------------------------;
                                ;39500          R[GPR.R+1]_RB.XZ,                 ; CLEAR BITS 31-16 (PL=0, SL=16).
U 0FE6, 0884,00B7,037F,C047,00F2,3  ;39501      STEPC.GE.4? DECBY4                ; LOOP ON THE STEP COUNTER.
```

```
                                        ;39502 =011
                                        ;39503          ;011-------------------------------;
U 0F23, 0080,0036,40B0,0047,03FF,F      ;39504          RETURN [-1]                       ;
                                        ;39505
                                        ;39506          ;111-------------------------------;
                                        ;39507          RNUM_R[TEMP0] RB+CONX(2),         ;
U 0F27, 0885,D73D,0010,0047,00FE,5      ;39508          NEXT7IE.CLR.GPRS15                ;
```

```
                                        ;39509 .TOC "  Interrupts and Exceptions       : Ird Rom Definition"
;39510  .NOBIN
;39511
;39512  .ICODE  ;       OPS             REG                     MEM                     OPS         FPA REG                 FPA MEM
;39513       OFD: FPD [NOP][FX.IE.FPD.RES      ]                                     [NOP][FX.IE.FPD.RES      ],                        ;ESCD
;39514            IRD1[NOP][FX.IE.FD.OPCOD     ]                                     [NOP][FX.IE.FD.OPCOD     ]
;39515  .OCODE
;39516       OFD: CNT0[NOP][FX.IE.FD.RET      ][FX.IE.FD.RET      ]  [NOP][FX.IE.FD.RET      ][FX.IE.FD.RET      ],
;39517            CNT1[NOP][FX.IE.FD.RET      ][FX.IE.FD.RET      ]  [NOP][FX.IE.FD.RET      ][FX.IE.FD.RET      ]
;39518
;39519  .ICODE
;39520       OFE: FPD [NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ],                        ;ESCE
;39521            IRD1[NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ]
;39522  .OCODE
;39523       OFE: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;39524            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;39525
;39526  .ICODE
;39527       OFF: FPD [NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ],                        ;ESCF
;39528            IRD1[NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ]
;39529  .OCODE
;39530       OFF: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;39531            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;39532
;39533  .ICODE
;39534       002: FPD [NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ],                        ;REI
;39535            IRD1[NOP][IE.REI            ]                                     [NOP][IE.REI            ]
;39536  .OCODE
;39537       002: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;39538            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;39539
;39540  .ICODE
;39541       057: FPD [NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ],                        ;RESVRD
;39542            IRD1[NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ]
;39543  .OCODE
;39544       057: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;39545            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;39546
;39547  .ICODE
;39548       059: FPD [NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ],                        ;RESVRD
;39549            IRD1[NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ]
;39550  .OCODE
;39551       059: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;39552            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;39553
;39554  .ICODE
;39555       05A: FPD [NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ],                        ;RESVRD
;39556            IRD1[NOP][IE.OPCOD.DEC      ]                                     [NOP][IE.OPCOD.DEC      ]
;39557  .OCODE
;39558       05A: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;39559            CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]  [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
```

```
;39560  .ICODE  ;      OPS          REG                    MEM                OPS        FPA REG                FPA MEM
;39561        05B: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;RESVRD
;39562             IRD1[NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ]
;39563  .OCODE
;39564        05B: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ] [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;39565             CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ] [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
;39566
;39567  .ICODE
;39568        077: FPD [NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ],                    ;RESVRD
;39569             IRD1[NOP][IE.OPCOD.DEC      ]                          [NOP][IE.OPCOD.DEC      ]
;39570  .OCODE
;39571        077: CNT0[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ] [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ],
;39572             CNT1[NOP][IE.BAD.IRD        ][IE.BAD.IRD        ] [NOP][IE.BAD.IRD        ][IE.BAD.IRD        ]
```

```
39573  .TOC "  Interrupts and Exceptions       : Dsize Rom Definition"
39574  .DCODE
39575
39576  0FD:    SIZE [DBLE] [DBLE] [DBLE] [DBLE] [DBLE] [DBLE]  ;ESCD
39577
39578  0FE:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;ESCE
39579
39580  0FF:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;ESCF
39581
39582  002:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;REI
39583
39584  057:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;RESVRD
39585
39586  059:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;RESVRD
39587
39588  05A:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;RESVRD
39589
39590  05B:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;RESVRD
39591
39592  077:    SIZE [   0] [   0] [   0] [   0] [   0] [   0]  ;RESVRD
39593
39594  .UCODE
                                    ;39595 .BIN
```

```
                                        ;39596 .TOC "MM.MIC"
                                        ;39597 .TOC "REVISION 30.0"
                                        ;39598 ;       Jeff Peng, Gerard Koeckhoven, Brian Allison, CHARLIE MCDOWELL
                                        ;39599
;39600          .NOBIN
;39601
;39602  .TOC "  Revision history"
;39603
;39604  ; REV   EXPLANATION
;39605
;39606  ; 30    Set flags in MM.GET.PTE and MM.GET.WRITE.PTE for TB parity error
;39607  ;       recovery.
;39608  ; 29    Set MM.NOINT when processing a page boundary write to avoid
;39609  ;       servicing an interrupt after writting the first page.
;39610  ;       used loc 1672
;39611  ; 28    Clear TB to avoid possible conflict between access violation
;39612  ;       and XB - TB-miss.
;39613  ; 27    Fix such that the VA will not be off by 4 when reading across
;39614  ;       a page boundry and FPD set and you restart the instruction.
;39615  ; 26    Fix such that the VA will not be off by 8 when writing crossing
;39616  ;       a page boundery and FPD set and you restart the instruction
;39617  ; 25    Change location 15FC to clear the TB  to avoid a conflict
;39618  ;       between an access violation and a xb-tb miss.
;39619  ; 24    Add label MM_PCALL_ACV_ENTRY for CALLx to jump to on ACV's
;39620  ; 23    Fix bug in MM.PRB.WRITE.SIZ if probe crosses a page boundary.
;39621  ;       Fix MM.PB.PROBE to properly restore MDR.
;39622  ;       Add BUS/PRB.RD.PTE to avoid machine hang loading the TB.
;39623  ; 22    Initial release.
                                        ;39624          .BIN
```

```
                                ;39625 .TOC "  Memory Management                : UNALIGNED READ"
                                ;39626
                                ;39627 ;*************************************************************************
                                ;39628 ;       UNALIGNED READ
                                ;39629 ;
                                ;39630 ;       Input           VA              Unaligned virtual address
                                ;39631 ;                       DSIZE signal    Read size
                                ;39632 ;
                                ;39633 ;       Resources       MM.TEMP1        Backup VA
                                ;39634 ;
                                ;39635 ;       Output          MDR             Completed read
                                ;39636 ;
                                ;39637 ;        CHARLIE'S FLOWS 1.4
                                ;39638 ;*************************************************************************
                                ;39639
                                ;39640 .REGION/MM.R1L,MM.R1H/MM.R2L,MM.R2H/MM.R3L,MM.R3H       ;GENERAL REGION
                                ;39641 21:     ;**UTRAP ADDR**
                                ;39642 MM.UA.READ:
                                ;39643         ;-------------------------------; FLUSH XB TO RESTART PRE-FETCH
U 0021, 0C81,A592,4030,0487,017F,A ;39644      FLUSH XB                        ; AND REMOVE NO BUS AND VA_X RESTRICTION
                                ;39645
                                ;39646 17FA:   ;*********FORCE ADDRESS*********;
                                ;39647         ;-------------------------------; SET FLAG TO CORRECT VA IF A FAULT,
                                ;39648         SET STACK FLAG,                 ; OCCURS ON THE READ.SECOND IN NEXT
U 17FA, 0C68,0036,4030,0442,0152,0 ;39649      READ.NOTRAP,VA_VA+4             ; CYCLE.  THIS IS NEEDED IF FPD=1
                                ;39650
                                ;39651 =0**    ;0**----------------------------;
                                ;39652         READ.SECOND,CLEAR STACK FLAG,   ; CLEAR FLAG IF READ IS SUCCESSFUL
U 1520, 0C28,0036,4030,0046,0176,8 ;39653      NEXT/MM.PB.WRITE15              ;
                                ;39654
                                ;39655         ;1**----------------------------; IF ACV/TNV WITH PSL<FPD>=1
U 1524, 0880,0036,40B0,0047,0000,4 ;39656      RETURN [+4]                     ; CATCH RETURN FROM FPDOFFSET=0
```

```
                                ;39657 .TOC "  Memory Management                : UNALIGNED WRITE, WRITE UNLOCK"
                                ;39658
                                ;39659 ;*****************************************************************************
                                ;39660 ;       UNALIGNED WRITE, WRITE UNLOCK.
                                ;39661 ;
                                ;39662 ;       Input           VA              Unaligned virtual address
                                ;39663 ;                       WDR             Data to be written
                                ;39664 ;
                                ;39665 ;       Resources       MM.TEMP1        Backup VA
                                ;39666 ;
                                ;39667 ;       CHARLIE'S FLOWS 1.4
                                ;39668 ;*****************************************************************************
                                ;39669
                                ;39670 24:     ;**UTRAP ADDR**
                                ;39671 MM.UA.WRITE.UNLOCK:
                                ;39672         ;-------------------------------;
                                ;39673         WRITE.NOTRAP,                   ;
U 0024, 0480,0036,4030,004C,0576,6 ;39674      PUSH,NEXT/MM.SAVE.INC.VA        ;
                                ;39675
                                ;39676 34:     ;-------------------------------;
                                ;39677         WRITE.SECOND.UL,                ;
U 0034, 0C80,0036,4030,004B,0176,7 ;39678      NEXT/MM.PB.WRITE10              ;
                                ;39679
                                ;39680 25:     ;**UTRAP ADDR**
                                ;39681 MM.UA.WRITE:
                                ;39682         ;-------------------------------;
                                ;39683         WRITE.NOTRAP,                   ;
U 0025, 0480,0036,4030,004C,0576,6 ;39684      PUSH,NEXT/MM.SAVE.INC.VA        ;
                                ;39685
                                ;39686 35:     ;-------------------------------;
                                ;39687         WRITE.SECOND,                   ;
U 0035, 0480,0036,4030,004A,0176,7 ;39688      NEXT/MM.PB.WRITE10              ;
                                ;39689
                                ;39690 MM.SAVE.INC.VA:
                                ;39691         ;-------------------------------;
                                ;39692         R[MM.TEMP1]_M[VA],VA_VA+4,      ; SAVE OLD VA, AND INCREMENT FOR
U 1766, 0085,B592,40B3,C447,0001,0 ;39693      RETURN [+10]                    ; SECOND HALF OF WRITE.
```

```
                                ;39694 .TOC "  Memory Management                : CROSSING PAGE BOUNDARY WRITE, WRITE UNLOCK
                                ;39695
                                ;39696 ;*************************************************************************
                                ;39697 ;       PAGE BOUNDARY WRITE, WRITE UNLOCK
                                ;39698 ;
                                ;39699 ;       Input           VA              Crossing page boundary virtual address
                                ;39700 ;                       WDR             Data to be written
                                ;39701 ;                       DSIZE signal    Write size
                                ;39702 ;
                                ;39703 ;       Resources       MM.TEMP4        Save WDR
                                ;39704 ;
                                ;39705 ;       Subroutines     MM.PB.PROBE
                                ;39706 ;                       MM.UA.WRITE
                                ;39707 ;                       MM.UA.WRITE.UNLOCK
                                ;39708 ;
                                ;39709 ;       CROSSING PAGE BOUNDARY WRITE, AND WRITE UNLOCK PUSH VA INTO THE NEXT
                                ;39710 ;       PAGE AND CALL MM.PB.PROBE TO CHECK THAT PAGE FOR WRITE ACCESS.
                                ;39711 ;       IF THE ACCESS IS OK, MM.PB.PROBE WILL RETURN-2 WHICH WILL
                                ;39712 ;       AUTOMATICALLY TRANSFER CONTROL TO MM.UA.WRITE OR MM.UA.WRITE.UNLOCK
                                ;39713 ;       WHICH ACTUALLY DOES THE WRITE.  IF THERE IS A TNV OR ACV, MM.PB.PROBE
                                ;39714 ;       WILL NOT RETURN.
                                ;39715 ;
                                ;39716 ;       CHARLIE'S FLOWS 1.4
                                ;39717 ;*************************************************************************
                                ;39718
                                ;39719 26:     ;**UTRAP ADDR**
                                ;39720 MM.PB.WRITE.UNLOCK:
                                ;39721         ;-------------------------------;
U 0026, 0485,3592,403B,C4C7,015E,C ;39722      R[MM.TEMP4]_WDR                 ; SAVE WDR.
                                ;39723
                                ;39724 =0      ;0-------------------------------;
                                ;39725         VA_M[VA]+ZLIT0[8],              ; LOAD AN ADDRESS IN THE NEXT PAGE.
U 15EC, 0581,BC11,0030,44A7,0533,D ;39726      PUSH,NEXT/MM.PB.PROBE           ; WILL RETURN-2 TO MM.UA.WRITE.UNLOCK
                                ;39727
                                ;39728         ;-------------------------------;
                                ;39729         WDR_UNROT(R[MM.TEMP4]),         ; RESTORE WDR.
                                ;39730         WRITE.SECOND,                   ; FINISH THE BUS CYCLE.
U 15ED, 0080,05BE,403B,C54A,0133,9 ;39731      NEXT/MM.PB.WRITE5               ; Modified for cmt066
```

```
                                ;39732 27:      ;**UTRAP ADDR**
                                ;39733
                                ;39734 MM.PB.WRITE:
                                ;39735          ;-------------------------------;
U 0027, 0485,3592,403B,C4C7,015F,4 ;39736       R[MM.TEMP4]_WDR         ; SAVE WDR.
                                ;39737
                                ;39738 =0       ;0------------------------------;
                                ;39739          VA_M[VA]+ZLIT0[8],              ; LOAD AN ADDRESS IN THE NEXT PAGE.
U 15F4, 0581,BC11,0030,44A7,0533,D ;39740       PUSH,NEXT/MM.PB.PROBE           ; WILL RETURN-2 TO MM.UA.WRITE
                                ;39741
                                ;39742          ;1------------------------------;
                                ;39743          WDR_UNROT(R[MM.TEMP4]),         ; RESTORE WDR.
                                ;39744          WRITE.SECOND,                   ; FINISH THE BUS CYCLE.
U 15F5, 0080,05BE,403B,C54A,0133,9 ;39745       NEXT/MM.PB.WRITE5               ; Modified for cmt066
                                ;39746
                                ;39747 ;1048:   ;++++++++ FORCE ADDRESS +++++++++
                                ;39748 ;MM.PB.WRITE5:
                                ;39749 ;        ;-------------------------------;
                                ;39750 ;        FLAGS_R[RTMPGPR],               ; RESTORE FLAGS
                                ;39751 ;        NEXT/MM.PB.WRITE10              ;
                                ;39752
                                ;39753 MM.PB.WRITE10:
                                ;39754          ;-------------------------------;
U 1767, 0481,A002,403D,8487,0176,8 ;39755       PC_M[PC]                        ; FLUSH THE XB TO REENABLE PRE-FETCH
                                ;39756
                                ;39757 MM.PB.WRITE15:
                                ;39758          ;-------------------------------;
                                ;39759          VA_M[VA]-ZLIT0[4],              ;
U 1768, 09D9,BC10,00B0,24A7,0000,0 ;39760       RETURN AND SUPPRESS BUS CYCLE   ;
```

```
                                ;39761 .TOC "  Memory Management                 :       MM.PB.PROBE"
                                ;39762
                                ;39763 ;*****************************************************************************
                                ;39764 ;       MM.PB.PROBE
                                ;39765 ;
                                ;39766 ;       Input           VA              Virtual address to be probed
                                ;39767 ;
                                ;39768 ;       Output          VA              VA - 8
                                ;39769 ;
                                ;39770 ;       Resources       MM.TEMP1        Save VA if TB Miss
                                ;39771 ;                       MM.TEMP3        Save faulting address if ACV or TNV
                                ;39772 ;                       MM.TEMP4        Save WDR
                                ;39773 ;
                                ;39774 ;       Subroutines     MM.GET.WRITE.PTE
                                ;39775 ;                       IE.TNV
                                ;39776 ;                       IE.ACV
                                ;39777 ;
                                ;39778 ;       MM.PB.PROBE IS USED TO CHECK THE ACCESS OF THE NEW PAGE IN A
                                ;39779 ;       CROSSING PAGE BOUNDARY MICRO TRAP.  IF THE ACCESS IS OK VA <- (VA-8)
                                ;39780 ;       AND IT DOES A RETURN-2.  IF THERE IS AN ACV OR TNV THE FAULT IS
                                ;39781 ;       INITIATED AND NO RETURN IS TAKEN.
                                ;39782 ;
                                ;39783 ;       CHARLIE'S FLOWS 1.4
                                ;39784 ;*****************************************************************************
                                ;39785 ;104A:
                                ;39786 ;MM.PB.PROBE:
                                ;39787 ;       ;***** FORCE LOCATION **********
                                ;39788 ;       ;------------------------------;
                                ;39789 ;       PROBE WRITE?,SIZE[BYTE],        ; PROBE VA USING CURRENT MODE.
                                ;39790 ;       R[RTMPGPR] FLAGS,               ; SAVE THE CURRENT EVENT FLAGS
                                ;39791 ;       NEXT/OLD16TC                    ; MAKE SURE WE SKIP OLD 1769
                                ;39792
                                ;39793 1769:
                                ;39794 FREE.1769:
                                ;39795         ;------------------------------;
U 1769, 0480,0036,4780,005D,0961,C 479*         ;39796         PROBE WRITE?,SIZE[BYTE]         ; PROBE VA USING CURRENT MODE.
                                ;39797
                                ;39798 161C:
                                ;39799 OLD161C:                                 ; HOPE IT REMAINS AT LOC 161C
                                ;39800         ;1100----------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                ;39801         R[MM.TEMP1] M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 161C, 0485,B592,4033,C047,0561,8              ;39802         PUSH,NEXT/MM.PRB.WRT.TBM        ; LOAD THE PTE BACK IN THE TB.
                                ;39803
                                ;39804 161D:   ;1101----------------------------; CATCH RETURN+1 FROM MM.GET.WRITE.PTE
                                ;39805         VA_M[VA]-ZLIT0[8],              ; DECREMENT VA BY 8
U 161D, 0181,BC10,0030,44A7,0158,2              ;39806         NEXT/MM.XB.TNV                  ; TAKE A TNV FAULT.
                                ;39807
                                ;39808 161E:   ;1110----------------------------; PROBE NO ACCESS OR RETURN+2 FROM
                                ;39809         R[MM.TEMP3] M[VA],              ; MM.GET.WRITE.PTE, TAKE AN ACV FAULT.
U 161E, 0485,B592,4039,C047,017F,6              ;39810         NEXT/MM.XB.ACV.-1               ;
                                ;39811
                                ;39812 161F:   ;1111----------------------------; ACCESS IS OK.
U 161F, 0981,BC10,0030,44A7,0176,A              ;39813         VA_M[VA]-ZLIT0[8]               ; PUT VA BACK IN FIRST PAGE.
                                ;39814
                                ;39815         ;------------------------------;
```

```
                                ;39816 176A:    WDR_UNROT(R[MM.TEMP4]),         ; RESTORE WDR.
U 176A, 0080,05BE,403B,C54C,0139,0  ;39817          WRITE.NOTRAP                    ; DO FIRST HALF OF THE BUS CYCLE.
                                ;39818
                                ;39819 1390:    ;+++++++ FORCE ADDRESS +++++++++; REV94
                                ;39820          ;------------------------------;
                                ;39821          VA_VA+4,SET MM.NOINT,           ; INC VA FOR SECOND HALF OF WRITE.
U 1390, 0860,0036,40B0,0447,0000,1  ;39822          RETURN [+1]                     ; DON'T ALLOW INTERUPTS NOW
                                ;39823
                                ;39824 176B:                                    ; ALL THIS SO WE CAN SAVE ONE ROM
                                ;39825 FREE.176B:                               ;
                                ;39826          ;*******FORCE ADDRESS**********;
                                ;39827          ;------------------------------;
U 176B, 0080,0036,40B0,0447,0000,1  ;39828          VA_VA+4,RETURN [+1]             ; INC VA FOR SECOND HALF OF WRITE.
                                ;39829
                                ;39830 17F6:
                                ;39831 MM.XB.ACV.-1:
                                ;39832          ;*******FORCE ADDRESS**********;
                                ;39833          VA_M[VA]-ZLIT0[8],              ; DECREMENT VA BY 8
U 17F6, 0981,BC10,0030,44A7,0158,3  ;39834          NEXT/MM.XB.ACV                  ;
```

```
                                ;39835 .TOC "  Memory Management                 : TB MISS"
                                ;39836
                                ;39837 ;*****************************************************************************
                                ;39838 ;       Entry points    MM.XB.TBMISS
                                ;39839 ;                       MM.READ.TBMISS
                                ;39840 ;                       MM.WRITE.TBMISS
                                ;39841 ;                       MM.BUT.XB.TBMISS
                                ;39842 ;
                                ;39843 ;       Input           VA      Faulting address for READ or WRITE TB MISSES
                                ;39844 ;                       PC      Faulting address for XB or BUT XB TB MISSES
                                ;39845 ;
                                ;39846 ;       Resources       MM.TEMP1        Save VA
                                ;39847 ;                       MM.TEMP2        Save MDR
                                ;39848 ;                       MM.TEMP4        Save WDR
                                ;39849 ;
                                ;39850 ;       Subroutines     MM.GET.XB.PTE
                                ;39851 ;                       MM.GET.READ.PTE
                                ;39852 ;                       MM.GET.WRITE.PTE
                                ;39853 ;                       MM.GET.BUT.PTE
                                ;39854 ;                       IE.TNV
                                ;39855 ;                       IE.ACV
                                ;39856 ;
                                ;39857 ;       THERE ARE 4 TB MISS MICRO TRAPS. THREE OF THEM (XB TB MISS,
                                ;39858 ;       READ TB MISS, AND WRITE TB MISS) START HERE BY MERELY
                                ;39859 ;       SAVING VA AND JUMPING TO A SUBROUTINE WHICH FINISHES PROCESSING
                                ;39860 ;       THE MISS AND RETURNS TO THREE DIFFERENT LOCATIONS, ONE FOR ACV, ONE
                                ;39861 ;       FOR TNV AND ONE IF THE MAPPING AND LOADING OF THE TB WAS SUCCESSFUL.
                                ;39862 ;       BUT XB TB MISS ALSO STARTS HERE BUT IT IS NOT PROCESSED BY AN
                                ;39863 ;       ACTUAL SUBROUTINE, IT MERELY SAVES VA AND PASSES CONTROL TO
                                ;39864 ;       ANOTHER BLOCK OF CODE WHICH WILL DO THE MAPPING AND RETURN CONTROL
                                ;39865 ;       TO THE TRAPPED INSTRUCTION.
                                ;39866 ;
                                ;39867 ;       CHARLIE'S FLOWS 1.1
                                ;39868 ;*****************************************************************************
                                ;39869
                                ;39870
                                ;39871 22:     ;**UTRAP ADDR**
                                ;39872 MM.XB.TBMISS:
                                ;39873         ;-------------------------------;
                                ;39874         R[MM.TEMP1]_M[VA],              ; SAVE VA.
U 0022, 0C85,B592,4033,C047,0158,1      ;39875         NEXT/MM.XB.TBMISS.10            ;
                                ;39876
                                ;39877 1580:   ;*********FORCE ADDRESS*********;
                                ;39878         ;000-----------------------------;
U 1580, 0C80,0036,4030,0047,0158,1      ;39879         NOP                             ; CATCH RETURN-1 FROM IE.SERV.IP.TS
                                ;39880
                                ;39881 1581:   ;*********FORCE ADDRESS*********;
                                ;39882 MM.XB.TBMISS.10:
                                ;39883         ;001-----------------------------;
                                ;39884         VA_M[PC]+R[ZERO],               ; LOAD VA WITH FAULTING ADDRESS.
U 1581, 0881,A001,003D,84A7,0576,C      ;39885         PUSH,NEXT/MM.GET.XB.PTE         ;
```

```
                                ;39886 1582:   ;*********FORCE ADDRESS*********;
                                ;39887 MM.XB.TNV:
                                ;39888         ;010--------------------------; CANNOT MAP, TAKE A TNV FAULT
                                ;39889         MDR_R[MM.TEMP2],              ; RESTORE MDR FOR PACKING
U 1582, 0880,05BE,4039,8467,00FB,5 ;39890      NEXT/IE.TNV.CHK.CONSOLE       ;
                                ;39891
                                ;39892 1583:   ;*********FORCE ADDRESS*********;
                                ;39893 MM.XB.ACV:
                                ;39894         ;011--------------------------; NO ACCESS, TAKE AN ACV FAULT
                                ;39895         MDR_R[MM.TEMP2],              ; RESTORE MDR FOR PACKING
U 1583, 0880,05BE,4039,8467,00FB,0 ;39896      NEXT/IE.ACV.CHK.CONSOLE       ;
                                ;39897
                                ;39898 1584:   ;100--------------------------; RESTORE MDR AND
U 1584, 0480,05BE,40B9,8467,0000,0 ;39899      MDR_R[MM.TEMP2],RETURN [+0]   ; REDO THE MICRO TRAPPED INSTRUCTION
                                ;39900
                                ;39901 1585:   ;101--------------------------; RETURN FROM IE.SERV.IP.TS
                                ;39902         MDR_R[MM.TEMP2],              ; RESTORE MDR
U 1585, 0C80,05BE,40B9,8467,0000,4 ;39903      RETURN [+4]                   ; AND TAKE SPECIAL FPD RETURN.
                                ;39904
                                ;39905 2A:     ;**UTRAP ADDR**
                                ;39906 MM.READ.TBMISS:
                                ;39907         ;-----------------------------;
                                ;39908         R[MM.TEMP1]_M[VA],            ; SAVE VA.
U 002A, 0485,B592,4033,C047,0159,1 ;39909      NEXT/MM.READ.TBMISS.10        ;
                                ;39910
                                ;39911 1590:   ;*********FORCE ADDRESS*********;
                                ;39912         ;000--------------------------;
U 1590, 0480,0036,4030,0047,0159,1 ;39913      NOP                           ; CATCH RETURN-1 FROM IE.SERV.IP.TS
                                ;39914
                                ;39915 1591:   ;*********FORCE ADDRESS*********;
                                ;39916 MM.READ.TBMISS.10:
                                ;39917         ;001--------------------------;
                                ;39918         R[MM.TEMP2]_M[MDR],           ; SAVE MDR.
                                ;39919         MM.ALLOW.INT?,                ;
U 1591, 0485,2592,42F9,8047,055C,0 ;39920      PUSH,NEXT/MM.GET.READ.PTE     ;
                                ;39921
                                ;39922 1592:   ;010--------------------------; CANNOT MAP, TAKE A TNV FAULT
                                ;39923         MEMSCAR_ZLIT24[1],            ; PREPARE TO FETCH READ MODIFY BIT
                                ;39924         STACK F[AG?,                  ; CHECK IF READ SEC FROM UNALIGNED READ
U 1592, 0980,0CB7,06B0,0E87,003F,C ;39925      NEXT/MM.READ.TNV              ;
                                ;39926 1593:
                                ;39927 MM.READ.NO.ACCESS:
                                ;39928         ;*********FORCE ADDRESS*********;
                                ;39929         ;011--------------------------; NO ACCESS, TAKE AN ACV FAULT
                                ;39930         MEMSCAR_ZLIT24[1],            ; PREPARE TO FETCH READ MODIFY BIT
                                ;39931         STACK F[AG?,                  ; CHECK IF READ SEC FROM UNALIGNED READ
U 1593, 0180,0CB7,06B0,0E87,003F,E ;39932      NEXT/MM.READ.ACV              ;
                                ;39933
                                ;39934 1594:   ;100--------------------------; MAPPING COMPLETED OK. RESTORE MDR AND
U 1594, 0480,05BE,40B9,8467,0000,0 ;39935      MDR_R[MM.TEMP2],RETURN [+0]   ; REDO THE MICRO TRAPPED INSTRUCTION
                                ;39936
                                ;39937 1595:   ;101--------------------------; RETURN FROM IE.SERV.IP.TS
                                ;39938         MDR_R[MM.TEMP2],              ; RESTORE MDR
U 1595, 0C80,05BE,40B9,8467,0000,4 ;39939      RETURN [+4]                   ; AND TAKE SPECIAL FPD RETURN.
```

```
                                ;39940 3FC:    ;*********FORCE ADDRESS*********;
                                ;39941 MM.READ.TNV:
                                ;39942         ;0-------------------------------; GET READ/MODIFY BIT TO SEPERATE
                                ;39943         R[MM.TEMP5] MEMSCR,               ; READ/MODIFY FROM STRAIGHT READS.
U 03FC, 0484,0036,4033,8647,00FA,5 ;39944      NEXT/IE.READ.TNV                  ;
                                ;39945
                                ;39946 3FD:    ;1********FORCE ADDRESS*********;
                                ;39947         CLEAR STACK FLAG,                 ; CLEAR STACK FLAG BEFORE FAULTING
                                ;39948         VA M[VA]-ZLIT0[4],                ; THIS IS FROM A READ SECOND REFERENCE
U 03FD, 0D29,BC10,0030,24A7,003F,C ;39949      NEXT/MM.READ.TNV                  ; NEED TO FIX VA BEFORE FAULTING
                                ;39950
                                ;39951 3FE:    ;*********FORCE ADDRESS*********;
                                ;39952 MM.READ.ACV:
                                ;39953         ;0-------------------------------; GET READ/MODIFY BIT TO SEPERATE
                                ;39954         R[MM.TEMP1] MEMSCR,               ; READ/MODIFY FROM STRAIGHT READS.
U 03FE, 0884,0036,4033,C647,00FA,4 ;39955      NEXT/IE.READ.ACV                  ;
                                ;39956
                                ;39957 3FF:    ;1********FORCE ADDRESS*********;
                                ;39958         CLEAR STACK FLAG,                 ; CLEAR STACK FLAG BEFORE FAULTING
                                ;39959         VA M[VA]-ZLIT0[4],                ; THIS IS FROM A READ SECOND REFERENCE
U 03FF, 0529,BC10,0030,24A7,003F,E ;39960      NEXT/MM.READ.ACV                  ; NEED TO FIX VA BEFORE FAULTING
```

```
                                ;39961 2B:      ;**UTRAP ADDR**
                                ;39962 MM.WRITE.TBMISS:
                                ;39963          ;-------------------------------;
                                ;39964          R[MM.TEMP1] M[VA],              ; SAVE VA.
U 002B, 0485,B592,4033,C047,015A,1 ;39965       NEXT/MM.WRITE.TBMISS.10         ;
                                ;39966
                                ;39967 15A0:    ;*********FORCE ADDRESS*********;
                                ;39968          ;000----------------------------;
U 15A0, 0480,0036,4030,0047,015A,1 ;39969       NOP                             ; CATCH RETURN-1 FROM IE.SERV.IP.TS
                                ;39970
                                ;39971 15A1:    ;*********FORCE ADDRESS*********;
                                ;39972 MM.WRITE.TBMISS.10:
                                ;39973          ;001----------------------------;
                                ;39974          R[MM.TEMP2] MLMDR],             ; SAVE MDR.
                                ;39975          MM.ALLOW.INT?,                  ;
U 15A1, 0C85,2592,42F9,8047,055E,0 ;39976       PUSH,NEXT/MM.GET.WRITE.PTE      ;
                                ;39977
                                ;39978 15A2:    ;010----------------------------; CANNOT MAP, TAKE A TNV FAULT
                                ;39979          MDR_R[MM.TEMP2],                ; RESTORE MDR FOR PACKING
U 15A2, 0880,05BE,4039,8467,00FB,5 ;39980       NEXT/IE.TNV.CHK.CONSOLE         ;
                                ;39981
                                ;39982 15A3:    ;011----------------------------; NO ACCESS, TAKE AN ACV FAULT
                                ;39983          MDR_R[MM.TEMP2],                ; RESTORE MDR FOR PACKING
U 15A3, 0880,05BE,4039,8467,00FB,0 ;39984       NEXT/IE.ACV.CHK.CONSOLE         ;
                                ;39985
                                ;39986 15A4:    ;100----------------------------; MAPPING COMPLETED OK. RESTORE MDR AND
U 15A4, 0480,05BE,40B9,8467,0000,0 ;39987       MDR_R[MM.TEMP2],RETURN [+0]     ; REDO THE MICRO TRAPPED INSTRUCTION
                                ;39988
                                ;39989 15A5:    ;101----------------------------; RETURN FROM IE.SERV.IP.TS
                                ;39990          MDR_R[MM.TEMP2],                ; RESTORE MDR
U 15A5, 0C80,05BE,40B9,8467,0000,4 ;39991       RETURN [+4]                     ; AND TAKE SPECIAL FPD RETURN.
                                ;39992
                                ;39993 29:      ;**UTRAP ADDR**
                                ;39994 MM.BUT.XB.TBMISS:
                                ;39995          ;-------------------------------;
                                ;39996          R[MM.TEMP1] M[VA],              ; SAVE VA
                                ;39997          OTHER UTRAPS?,                  ; PATCH TU DETECT UTRAPS IN WRONG ORDER.
U 0029, 0885,B592,4833,C047,015F,C ;39998       NEXT/MM.GET.BUT.PTE             ;
```

```
                        ;39999 .TOC "  Memory Management                : MM.GET.XB.PTE, MM.GET.READ.PTE"
                        ;40000
                        ;40001 ;*****************************************************************************
                        ;40002 ;       Entry points    MM.GET.XB.PTE
                        ;40003 ;                       MM.GET.READ.PTE
                        ;40004 ;
                        ;40005 ;       Input           VA              Faulting address
                        ;40006 ;
                        ;40007 ;       Resources       ERRCOD          Error code if ACV or TNV
                        ;40008 ;                       MM.TEMP2        Save MDR
                        ;40009 ;                       MM.TEMP3        Save faulting address
                        ;40010 ;
                        ;40011 ;       Subroutines     MM.GET.PTE
                        ;40012 ;                       IE.SERV.IP.TS
                        ;40013 ;
                        ;40014 ;       MM.GET.XB.PTE CHECKS FOR A TB MISS ACROSS A PAGE BOUNDARY, SAVES
                        ;40015 ;       THE PROPER FAULTING ADDRESS(PC OR PC+4) AND THEN JOINS THE FLOW
                        ;40016 ;       OF MM.GET.READ.PTE.
                        ;40017 ;
                        ;40018 ;       MM.GET.READ.PTE CALLS MM.GET.PTE WHICH RETURNS WITH THE NEW PTE IN
                        ;40019 ;       MDR OR ELSE TAKES A SEPERATE RETURN FOR ACV OR TNV.  THE NEW PTE
                        ;40020 ;       IS THEN CHECKED FOR VALIDITY AND READ ACCESS AND IF OK IT IS
                        ;40021 ;       LOADED INTO THE TB.
                        ;40022 ;
                        ;40023 ;       CHARLIE'S FLOWS 1.2
                        ;40024 ;*****************************************  *********************************
                        ;40025
                        ;40026 MM.GET.XB.PTE:
                        ;40027          ;-------------------------------;
U 176C, 0480,0036,4780,005F,0962,E 479* ;40028          PROBE READ?,SIZE[BYTE]          ; CHECK FOR MISS ACROSS A PAGE BOUNDARY.
                        ;40029
                        ;40030 =1110
                        ;40031 MM.GET.XB.PTE10:
                        ;40032          ;1110---------------------------;
                        ;40033          R[MM.TEMP2]_M[MDR],             ; SAVE MDR.
                        ;40034          MM.ALLOW.INT?,                  ;
U 162E, 0C85,2592,42F9,8047,015C,0      ;40035          NEXT/MM.GET.READ.PTE            ;
                        ;40036
                        ;40037          ;1111---------------------------;
                        ;40038          VA_VA+4,                        ; MUST BE A MISS ACROSS A PAGE BOUNDARY.
U 162F, 0085,2592,4039,8447,0176,D      ;40039          R[MM.TEMP2]_M[MDR]              ; SAVE MDR.
                        ;40040
                        ;40041          ;-------------------------------;
U 176D, 0C80,0036,4780,005F,0963,E 479* ;40042          PROBE READ?,SIZE[BYTE]          ; CHECK IF STILL A MISS IN NEW PAGE.
                        ;40043
                        ;40044 =1110    ;1110---------------------------;
                        ;40045          MM.ALLOW.INT?,                  ;
U 163E, 0880,0036,42F0,0047,015C,0      ;40046          NEXT/MM.GET.READ.PTE            ;
                        ;40047
                        ;40048          ;1111---------------------------; STALE XB FLAG, NO LONGER A MISS
                        ;40049          VA_R[MM.TEMP1],                 ; RESTORE VA AND
U 163F, 0080,05BE,40B3,C4A7,0000,3      ;40050          RETURN [+3]                     ; TAKE GOOD RETURN.
```

```
                                ;40051 =000
                                ;40052 MM.GET.READ.PTE:
                                ;40053          ;000----------------------------;
                                ;40054          R[MM.TEMP3]_M[VA],              ; SAVE FAULTING ADDRESS.
                                ;40055          WB<31-30>?,                     ;
U 15C0, 0485,8592,46F9,C047,055D,8 ;40056       PUSH.NEXT/MM.GET.PTE            ;
                                ;40057
                                ;40058 =010     ;010----------------------------;
                                ;40059          VA_R[MM.TEMP1],                 ; RESTORE VA FOR PACK ROUTINE
                                ;40060          INTPEND OR TIMER?,              ;
U 15C2, 0C80,05BE,4AF3,C4A7,00F7,0 ;40061       NEXT/IE.SERV.IP.TS2             ; SERVICE PENDING INTERRUPT OR TIMER.
                                ;40062 =100
                                ;40063 MM.GET.ACV:
                                ;40064          ;100----------------------------; NO ACCESS TRYING TO FETCH PTE,
                                ;40065          VA_R[MM.TEMP1],                 ; RESTORE VA AND
U 15C4, 0880,05BE,40B3,C4A7,0000,2 ;40066       RETURN [+2]                     ; RETURN INDICATING ACV FAULT.
                                ;40067
                                ;40068 MM.GET.TNV:
                                ;40069          ;101----------------------------; NOT VALID TRYING TO FETCH PTE,
                                ;40070          VA_R[MM.TEMP1],                 ; RESTORE VA AND
U 15C5, 0880,05BE,40B3,C4A7,0000,1 ;40071       RETURN [+1]                     ; RETURN INDICATING TNV FAULT.
                                ;40072
                                ;40073          ;110----------------------------; SUCCESSFULLY FETCHED THE NEW PTE.
                                ;40074          R[MM.TEMP5]_M[MDR].RL.9,        ; THE NEW PTE IS IN MDR,
                                ;40075          PTE CHECK READ?,                ; CHECK IT FOR READ ACCESS.
U 15C6, 0485,2477,07B3,8056,095D,4 479* ;40076  NEXT/MM.GET.READ.PTE20           ;
                                ;40077
                                ;40078          ;111----------------------------; SUCCESSFULLY FETCHED THE NEW PTE.
                                ;40079          R[MM.TEMP5]_M[MDR].RL.9,        ; THE NEW PTE IS IN MDR,
                                ;40080          PTE CHECK READ?,                ; CHECK IT FOR READ ACCESS.
U 15C7, 0485,2477,07B3,8056,095D,4 479* ;40081  NEXT/MM.GET.READ.PTE20           ;
                                ;40082
                                ;40083 =100
                                ;40084 MM.GET.READ.PTE20:
                                ;40085          ;100----------------------------; NO READ ACCESS.
                                ;40086          M[ERRCOD]_ZLIT0[0],             ; SET ERROR CODE AND
U 15D4, 0986,BC37,0030,0047,015C,4 ;40087       NEXT/MM.GET.ACV                 ; JUMP TO ACV RETURN.
                                ;40088
                                ;40089          ;101----------------------------; READ ACCESS OK BUT NOT VALID,
                                ;40090          M[ERRCOD]_ZLIT0[0],             ; SET ERROR CODE AND
U 15D5, 0186,BC37,0030,0047,015C,5 ;40091       NEXT/MM.GET.TNV                 ; TAKE TNV FAULT
                                ;40092
                                ;40093 =111     ;111----------------------------;
                                ;40094          COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG LOADING THE TB
                                ;40095          VA_R[MM.TEMP3],                 ; LOAD FAULTING ADDRESS INTO VA.
U 15D7, 0480,05BE,4039,C4B7,0177,0 ;40096       NEXT/MM.GET.WRITE.PTE40         ;
```

```
;40097 .TOC "  Memory Management                  : MM.GET.WRITE.PTE"
;40098
;40099 ;*************************************************************************
;40100 ;       Entry Points    MM.GET.WRITE.PTE
;40101 ;
;40102 ;       Input           VA                  Faulting address
;40103 ;
;40104 ;       Subroutines     MM.GET.PTE
;40105 ;                       IE.SERV.IP.TS
;40106 ;                       MM.WRITE.SPTE
;40107 ;
;40108 ;       Resources       MM.TEMP1            Save VA
;40109 ;                       MM.TEMP2            Save MDR
;40110 ;                       MM.TEMP3            Save faulting address
;40111 ;                       MM.TEMP5            Save new PTE rotated for TB loading
;40112 ;
;40113 ;       MM.GET.WRITE.PTE CALLS MM.GET.PTE TO FETCH THE PTE INTO MDR.
;40114 ;       MM.GET.PTE RETURNS 4 WAYS, ONE FOR ACV, ONE FOR TNV, ONE IF THE
;40115 ;       PTE IS IN MDR AND VA CONTAINS THE VIRTUAL ADDRESS OF THE PTE,
;40116 ;       AND ONE IF THE PTE IS IN MDR AND VA CONTAINS THE PHYSICAL ADDRESS
;40117 ;       OF THE PTE.  THESE LAST TWO ARE TO ALLOW MODIFYING OF THE PTE IN
;40118 ;       MEMORY IF PTE<M>=0.  ONCE THE PTE IS IN MDR IT IS CHECKED
;40119 ;       FOR WRITE ACCESS.  IF OK AND PTE<M>=1 THEN THE PTE IS LOADED INTO
;40120 ;       THE TB.  IF THERE IS AN ACV OR TNV THE WRITE BIT IS SET IN THE
;40121 ;       ERROR CODE AND THE FAULT IS INITIATED.
;40122 ;
;40123 ;       IF THE ACCESS IS OK AND PTE<M>=0 THEN THE PTE IS MODIFIED AND WRITTEN
;40124 ;       BACK TO MEMORY BEFORE LOADING IT INTO THE TB. IF PTE<M>=0 AND VA
;40125 ;       CONTAINS THE VIRTUAL ADDRESS OF THE PTE, IT IS NECESSARY TO
;40126 ;       CHECK THAT THE SYSTEM PTE FOR THAT PROCESS PTE HAS ITS MODIFY BIT
;40127 ;       SET. IF SPTE<M>=0 THEN THE ENTRY IN THE TB IS MODIFIED TO AVOID
;40128 ;       A WRITE M=0 MICRO TRAP WHEN WRITING THE PROCESS PTE OUT TO MEMORY.
;40129 ;
;40130 ;       CHARLIE'S FLOWS 1.2
;40131 ;*************************************************************************
;40132 ;       TB PARITY ERROR RECOVERY
;40133 ;
;40134 ;       Resources       FPDOFFSET           bit<29> as a flag of MM.GET.WRITE.PTE
;40135 ;                                           bus cycle in progress
;40136 ;*************************************************************************
;40137 =000
;40138 MM.GET.WRITE.PTE:
;40139                 ;000-------------------------------;
;40140                 R[MM.TEMP3]_M[VA],                ; SAVE FAULTING ADDRESS.
;40141                 WB<31-30>?,                       ;
U 15E0, 0485,B592,46F9,C047,055D,8  ;40142              PUSH,NEXT/MM.GET.PTE              ;
;40143
;40144 =010      ;010-------------------------------;
;40145           INTPEND OR TIMER?,                      ;
U 15E2, 0C80,0036,4AF0,0047,00F7,0  ;40146              NEXT/IE.SERV.IP.TS2               ; SERVICE PENDING INTERRUPT OR TIMER.
;40147
;40148 =100      ;100-------------------------------; NO ACCESS TRYING TO FETCH PTE,
;40149           M[ERRCOD]_MB.OR.ZLIT0[4],               ; SET WRITE BIT IN ERROR CODE AND
U 15E4, 0186,BC12,4030,2047,015C,4  ;40150              NEXT/MM.GET.ACV                   ; JUMP TO ACV RETURN.
;40151
```

```
                                ;40152          ;101----------------------------; NOT VALID TRYING TO FETCH PTE,
                                ;40153          M[ERRCOD] MB.OR.ZLIT0[4],       ; SET WRITE BIT IN ERROR CODE AND
U 15E5, 0986,BC12,4030,2047,015C,5 ;40154       NEXT/MM.GET.TNV                 ; TAKE TNV RETURN.
                                ;40155
                                ;40156          ;110----------------------------; SUCCESSFULLY FETCHED THE NEW PTE.
                                ;40157          R[MM.TEMP5] M[MDR].RL.9,        ; THE NEW PTE IS IN MDR,
                                ;40158          PTE CHECK WRITE?,               ; CHECK IT FOR WRITE ACCESS.
U 15E6, 0C85,2477,07B3,8052,095F,0 479* ;40159  NEXT/MM.GET.WRITE.PTE20          ; VA CONTAINS THE PA OF THE PTE.
                                ;40160
                                ;40161          ;111----------------------------; SUCCESSFULLY FETCHED THE NEW PTE.
                                ;40162          R[MM.TEMP5] M[MDR].RL.9,        ; THE NEW PTE IS IN MDR,
                                ;40163          PTE CHECK WRITE?,               ; CHECK IT FOR WRITE ACCESS.
U 15E7, 0C85,2477,07B3,8052,0960,0 479* ;40164  NEXT/MM.GET.WRITE.PTE30          ; VA CONTAINS THE VA OF THE PTE
                                ;40165 =000
                                ;40166 MM.GET.WRITE.PTE20:
                                ;40167          ;000----------------------------; NO WRITE ACCESS,
                                ;40168          M[ERRCOD] ZLIT0[4],             ; SET ERROR CODE AND
U 15F0, 0186,BC37,0030,2047,015C,4 ;40169       NEXT/MM.GET.ACV                 ; JUMP TO ACV RETURN.
                                ;40170
                                ;40171          ;001----------------------------; WRITE ACCESS OK BUT NOT VALID,
                                ;40172          M[ERRCOD] ZLIT0[4],             ; SET ERROR CODE AND
U 15F1, 0986,BC37,0030,2047,015C,5 ;40173       NEXT/MM.GET.TNV                 ; TAKE TNV RETURN.
                                ;40174
                                ;40175 =011
                                ;40176 MM.GET.WRITE.PTE25:
                                ;40177          ;011----------------------------; WRITE ACCESS OK AND VALID PTE BUT
                                ;40178          WRITE.PHY,                      ; M=0.
                                ;40179          WDR_UNROT(M[MDR].OR.ZLIT24[4]), ; WRITE PTE BACK TO MEMORY SETTING M=1.
U 15F3, 0981,2C92,4030,2548,0176,E ;40180       NEXT/MM.GET.WRITE.PTE33         ;
                                ;40181
                                ;40182 =111     ;111----------------------------;
                                ;40183          COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG LOADING THE TB
                                ;40184          VA_R[MM.TEMP3],                 ; RESTORE FAULTING ADDRESS.
U 15F7, 0480,05BE,4039,C4B7,0177,0 ;40185       NEXT/MM.GET.WRITE.PTE40         ;
                                ;40186
                                ;40187 =000
                                ;40188 MM.GET.WRITE.PTE30:
                                ;40189          ;000----------------------------; NO WRITE ACCESS,
                                ;40190          M[ERRCOD] ZLIT0[4],             ; SET ERROR CODE AND
U 1600, 0186,BC37,0030,2047,015C,4 ;40191       NEXT/MM.GET.ACV                 ; JUMP TO ACV RETURN.
                                ;40192
                                ;40193          ;001----------------------------; WRITE ACCESS OK BUT NOT VALID,
                                ;40194          M[ERRCOD] ZLIT0[4],             ; SET ERROR CODE AND
U 1601, 0986,BC37,0030,2047,015C,5 ;40195       NEXT/MM.GET.TNV                 ; TAKE TNV RETURN.
                                ;40196
                                ;40197 =011     ;011----------------------------; WRITE ACCESS OK AND VALID PTE BUT
                                ;40198          PROBE READ MODE ON WB?,         ; CHECK M-BIT IN SYSTEM PTE.
                                ;40199          SIZE[BYTE],WB_R[ZERO],          ;
                                ;40200          PATCH,                          ; ** jump to PCS **
U 1603, 1C80,05BE,478D,805E,0964,8 479* ;40201  NEXT/MM.GET.WRITE.PTE32         ;
                                ;40202
```

```
                                ;40203 =111
                                ;40204 MM.GET.WRITE.PTE31:
                                ;40205          ;111----------------------------;
                                ;40206          COMPLETE CPU BUS CYCLES,        ;AVOID MACHINE HANG LOADING THE TB
                                ;40207          VA_R[MM.TEMP3],                 ; RESTORE FAULTING ADDRESS.
U 1607, 0480,05BE,4039,C4B7,0177,0 ;40208       NEXT/MM.GET.WRITE.PTE40         ;
                                ;40209 =1011
                                ;40210 MM.GET.WRITE.PTE32:
                                ;40211          ;1011############################; ** code replaced in PCS **
                                ;40212          M[MM.TEMP0]_Q,                  ; M=0 IN SPTE, SAVE Q.
U 164B, 0086,D03A,403D,8047,0177,1 ;40213       NEXT/MM.SPTE.M.EQ.0             ;
                                ;40214
                                ;40215          ;1111############################; ** code replaced in PCS **
                                ;40216          WRITE.LONG.NOTRAP,              ; M=1 IN SPTE, M=0 IN PROCESS PTE.
                                ;40217          WDR_UNROT(M[MDR].OR.ZLIT24[4]), ; WRITE PTE BACK TO MEMORY SETTING M=1.
U 164F, 0981,2C92,4030,254E,0176,E ;40218       NEXT/MM.GET.WRITE.PTE33         ;
                                ;40219
                                ;40220 2A4B:    ;1011============================; ** code in PCS **
                                ;40221          M[MM.TEMP0]_Q,                  ; M=0 IN SPTE, SAVE Q.
U 2A4B, 0086,D03A,403D,8047,0177,1 ;40222       NEXT/MM.SPTE.M.EQ.0             ;
                                ;40223
                                ;40224 2A4F:    ;1111============================; ** code in PCS **
                                ;40225          M[FPDOFFSET]_MB.OR.ZLIT28[02],  ; M=1 IN SPTE
U 2A4F, 0D86,CC52,4030,1047,02A4,C ;40226       NEXT/MM.GET.WRITE.PCS33         ; SET MM PTE BUS CYCLE FLAG
                                ;40227
                                ;40228 2A4C:
                                ;40229 MM.GET.WRITE.PCS33:
                                ;40230          ;1100============================;
                                ;40231          WRITE.LONG.NOTRAP,              ; M=0 IN PROCESS PTE
U 2A4C, 0981,2C92,4030,254E,02A4,D ;40232       WDR_UNROT(M[MDR].OR.ZLIT24[4])  ; WRITE PTE BACK TO MEMORY SETTING M=1
                                ;40233                                          ; If TB parity error traped here, takes
                                ;40234                                          ; RETURN [-1] in recovery routine.
                                ;40235
                                ;40236 2A4D:    ;1101============================;
U 2A4D, 0186,CC53,8030,1047,0176,E ;40237       M[FPDOFFSET]_MB.ANDNOT.ZLIT28[2]; CLEAR BUS CYCLE FLAG
                                ;40238
                                ;40239 MM.GET.WRITE.PTE33:
                                ;40240          ;--------------------------------;
U 176E, 0181,2C92,4030,2467,0176,F ;40241       MDR_M[MDR].OR.ZLIT24[4]         ; SET M=1 IN NEW PTE.
                                ;40242
                                ;40243          ;--------------------------------;
                                ;40244          R[MM.TEMP5]_M[MDR].RL.9,        ; SAVE PTE ROTATED FOR LOADING INTO TB
U 176F, 0485,2477,0033,8047,0160,7 ;40245       NEXT/MM.GET.WRITE.PTE31         ;
                                ;40246
                                ;40247 MM.GET.WRITE.PTE40:                      ; MM.GET.READ.PTE JOINS FLOW HERE.
                                ;40248          ;--------------------------------;
                                ;40249          TB_R[MM.TEMP5],                 ; LOAD THE NEW PTE INTO THE TB.
U 1770, 0081,B5BE,4033,8507,0161,1 ;40250       NEXT/MM.GET.WRITE.PTE60         ;
                                ;40251
```

```
                                ;40252 =000
                                ;40253 MM.GET.WRITE.PTE55:
                                ;40254          ;000-------------------------------;
                                ;40255          PC_M[PC],                         ; FLUSH XB TO RESTORE PRE-FETCH
U 1610, 0881,A002,40BD,8487,0000,3 ;40256       RETURN [+3]                       ;
                                ;40257
                                ;40258 MM.GET.WRITE.PTE60:
                                ;40259          ;001-------------------------------; CATCH RETURN-1 FROM IE.SERV.IP.TS
                                ;40260          VA_R[MM.TEMP1],                   ; RESTORE VA AND CHECK
                                ;40261          MM.ALLOW.INT?,                    ; FOR INTERUPTS IF MM.NOINT IS CLEAR
U 1611, 0080,05BE,42F3,C4A7,0161,0 ;40262       NEXT/MM.GET.WRITE.PTE55           ;
                                ;40263
                                ;40264 =010     ;010-------------------------------;
                                ;40265          MDR_R[MM.TEMP2],                  ; RESTORE MDR FOR PACK ROUTINE.
                                ;40266          INTPEND OR TIMER?,                ;
U 1612, 0080,05BE,4AF9,8467,04F7,0 ;40267       PUSH,NEXT/IE.SERV.IP.TS2          ; SERVICE PENDING INTERRUPT OR TIMER.
                                ;40268
                                ;40269 =110     ;110-------------------------------; CATCH RETURN+4 FROM IE.SERV.IP.TS
U 1616, 0880,0036,40B0,0047,0000,4 ;40270       RETURN [+4]                       ; TAKE SPECIAL FPD RETURN.
                                ;40271 =
```

```
                                ;40272 .TOC "  Memory Management                : MM.SPTE.M.EQ.0"
                                ;40273
                                ;40274 ;*****************************************************************************
                                ;40275 ;       Entry points    MM.SPTE.M.EQ.0
                                ;40276 ;
                                ;40277 ;       Input           VA              Virtual Address of PTE (VAPTE)
                                ;40278 ;
                                ;40279 ;       Resources       MM.TEMP0        Save Q
                                ;40280 ;                       MM.TEMP5        Save the process PTE (was in MDR)
                                ;40281 ;                       MDR             Fetch system PTE
                                ;40282 ;                       Q               Hold physical address of SPTE
                                ;40283 ;                       LONLIT          Mask for PFN
                                ;40284 ;
                                ;40285 ;       MM.WRITE.SPTE IS CALLED ONLY WHEN WRITING A PROCESS PTE TO MEMORY
                                ;40286 ;       IN ORDER TO SET M=1 AND THE SYSTEM PTE FOR THAT PAGE OF PROCESS
                                ;40287 ;       PTE'S HAS M=0.
                                ;40288 ;       IN THIS CASE THE PHYSICAL ADDRESS OF THE PROCESS PTE IS CALCULATED
                                ;40289 ;       AND A WRITE PHYSICAL IS DONE TO SET THE M BIT IN THE PROCESS PTE.
                                ;40290 ;       THE SPTE IS NOT AFFECTED.
                                ;40291 ;
                                ;40292 ;       CHARLIE'S FLOWS 1.2.1
                                ;40293 ;*****************************************************************************
                                ;40294
                                ;40295 MM.SPTE.M.EQ.0:
                                ;40296         ;-------------------------------;
U 1771, 0481,B592,5030,0047,0177,2  ;40297     Q_M[VA]                         ; SAVE VAPTE.
                                ;40298
                                ;40299         ;-------------------------------;
U 1772, 0881,B33D,003B,04A7,0177,3  ;40300     VA_R[SBR]+M[VA].PTX             ; COMPUTE PA OF SPTE
                                ;40301
                                ;40302         ;-------------------------------;
U 1773, 0980,0CB7,0030,0687,0177,4  ;40303     MEMSCAR_ZLIT24[0]               ; LOAD ADDRESS OF MME BIT
                                ;40304
                                ;40305         ;-------------------------------;
U 1774, 0080,05B7,0030,0607,0177,5  ;40306     MEMSCR_0                        ; TURN OFF MEMORY MANAGEMENT
                                ;40307
                                ;40308         ;-------------------------------;
U 1775, 0181,BC13,8030,1CA7,0177,6  ;40309     VA_M[VA].ANDNOT.ZLIT0[3]        ; IGNORE LOW 2 BITS OF ADDRESS.
                                ;40310
                                ;40311         ;-------------------------------; READ.PHY GRONKS 2ND MDR SO, MM IS
                                ;40312         READ.NOTRAP,                    ; TURNED OFF TO DO A PHYSICAL READ.
U 1776, 0085,2592,4033,8042,0177,8  ;40313     R[MM.TEMP5]_M[MDR]              ; FETCH THE SPTE.  SAVE THE PROCESS PTE.
                                ;40314
                                ;40315         ;-------------------------------;
U 1778, 0480,0E77,0030,0607,0177,9  ;40316     MEMSCR_-1                       ; TURN MEMORY MANAGEMENT BACK ON.
                                ;40317
                                ;40318         ;-------------------------------;
U 1779, 0C81,2477,0030,0467,0177,A  ;40319     MDR_M[MDR].RL.9                 ;
```

```
                                ;40320  ;--------------------------------;
U 177A, 0480,003A,403D,84A7,0177,B ;40321  VA_Q                            ; RESTORE VAPTE.
                                ;40322
                                ;40323  ;--------------------------------;
U 177B, 0D81,2C13,903F,F847,0177,C ;40324  Q_M[MDR].ANDNOT.ZLIT0[1FF]       ; COMPUTE PAGE PORTION OF ADDRESS.
                                ;40325
                                ;40326  ;--------------------------------;
U 177C, 0181,BC12,003F,FCA7,0177,D ;40327  VA_M[VA].AND.ZLIT0[1FF]          ; COMPUT BYTE PORTION OF ADDRESS.
                                ;40328
                                ;40329  ;--------------------------------;
U 177D, 0481,B00A,4030,04A7,0177,E ;40330  VA_M[VA].OR.Q                    ; LOAD COMPLETE PA OF PROCESS PTE.
                                ;40331
                                ;40332  ;--------------------------------;
U 177E, 0880,D004,7033,8467,0178,0 ;40333  Q_M[MM.TEMP0] MDR_R[MM.TEMP5]   ; RESTORE Q AND RESTORE PTE TO MDR.
                                ;40334
                                ;40335  ;--------------------------------;
                                ;40336  R[MM.TEMP5]_M[MDR].RL.9,           ; RESTORE MM.TEMP5 (ROTATED PTE).
U 1780, 0C85,2477,0033,8047,015F,3 ;40337  NEXT/MM.GET.WRITE.PTE25          ;
```

```
                                ;40338 .TOC "  Memory Management                : MM.GET.BUT.PTE"
                                ;40339
                                ;40340 ;*****************************************************************************
                                ;40341 ;       Entry points    MM.GET.BUT.PTE
                                ;40342 ;
                                ;40343 ;       Input           PC              Faulting address
                                ;40344 ;
                                ;40345 ;       Resources       ERRCOD          Error code if ACV or TNV
                                ;40346 ;                       MM.TEMP1        Save VA
                                ;40347 ;                       MM.TEMP2        Save MDR
                                ;40348 ;                       MM.TEMP3        Save faulting address
                                ;40349 ;
                                ;40350 ;       Subroutines     MM.GET.PTE
                                ;40351 ;                       IE.SERV.IP.TS
                                ;40352 ;
                                ;40353 ;       MM.GET.BUT.PTE CHECKS FOR A TB MISS ACROSS A PAGE BOUNDARY, SAVES
                                ;40354 ;       THE PROPER FAULTING ADDRESS(PC OR PC+4) AND THEN
                                ;40355 ;       CALLS MM.GET.PTE WHICH RETURNS WITH THE NEW PTE IN
                                ;40356 ;       MDR OR ELSE TAKES A SEPERATE RETURN FOR ACV OR TNV.  THE NEW PTE
                                ;40357 ;       IS THEN CHECKED FOR VALIDITY AND READ ACCESS AND IF OK IT IS
                                ;40358 ;       LOADED INTO THE TB AND THE TRAPPED INSTRUCTI  I IS RE-EXECUTED.
                                ;40359 ;
                                ;40360 ;       CHARLIE'S FLOWS 1.2
                                ;40361 ;*****************************************************************************
                                ;40362 =0
                                ;40363 MM.GET.BUT.P  .:
                                ;40364          ;0------------------------------;
U 15FC, 0881,B5BE,40BD,8507,0000,0  ;40365          RETURN [+0],TB_R[ZERO]          ; SOME OTHER UTRAP NEEDS SERVICE FIRST.
                                ;40366
                                ;40367          ;1------------------------------;
U 15FD, 0081,A001,003D,84A7,0178,1  ;40368          VA_M[PC]+R[ZERO]                ; LOAD VA WITH FAULTING ADDRESS.
                                ;40369
                                ;40370          ;------------------------------;
U 1781, 0C80,0036,4780,005F,0965,E 479*  ;40371          PROBE READ?,SIZE[BYTE]          ; CHECK FOR MISS ACROSS A PAGE BOUNDARY.
                                ;40372 =1110
                                ;40373 MM.GET.BUT.PTE05:
                                ;40374          ;1110-----------------------------;
                                ;40375          R[MM.TEMP2]_M[MDR],             ; SAVE MDR.
                                ;40376          MM.ALLOW.INT?,                  ;
U 165E, 0485,2592,42F9,8047,0162,0  ;40377          NEXT/MM.GET.BUT.PTE10           ;
                                ;40378
                                ;40379          ;1111-----------------------------; MUST BE A MISS ACROSS A PAGE BOUNDARY.
U 165F, 0885,2592,4039,8447,0178,2  ;40380          R[MM.TEMP2]_M[MDR].VA_VA+4      ; SAVE MDR.
                                ;40381
                                ;40382          ;------------------------------;
U 1782, 0C80,0036,4780,005F,0966,E 479*  ;40383          PROBE READ?,SIZE[BYTE]          ; CHECK IF STILL A MISS IN NEW PAGE.
                                ;40384
                                ;40385 =1110    ;1110-----------------------------;
                                ;40386          MM.ALLOW.INT?,                  ;
U 166E, 0080,0036,42F0,0047,0162,0  ;40387          NEXT/MM.GET.BUT.PTE10           ;
```

```
                                ;40388 MM.GET.BUT.FLUSH:
                                ;40389          ;1111-------------------------; STALE XB FLAG, NO LONGER A MISS
U 166F, 0481,A592,4030,0487,0178,3 ;40390       FLUSH XB                        ;
                                ;40391
                                ;40392          ;-------------------------------;
                                ;40393          VA_R[MM.TEMP1],                 ; RESTORE VA,
U 1783, 0480,05BE,40F3,C4A7,0000,0 ;40394       RETURN AND INHIBIT DESTINATIONS,NEXT/0 ;
                                ;40395 =000
                                ;40396 MM.GET.BUT.PTE10:
                                ;40397          ;000--------------------------;
                                ;40398          R[MM.TEMP3]_M[VA],              ; SAVE FAULTING ADDRESS.
                                ;40399          WB<31-30>?,                     ;
U 1620, 0485,B592,46F9,C047,055D,8 ;40400       PUSH,NEXT/MM.GET.PTE            ;
                                ;40401
                                ;40402          ;001--------------------------; CATCH RETURN-1 FROM IE.SERV.IP.TS
U 1621, 0480,0036,4030,0047,0162,0 ;40403       NEXT/MM.GET.BUT.PTE10           ; CONTINUE AS IF NO INTERRUPT
                                ;40404
                                ;40405          ;010--------------------------;
                                ;40406          M[FPDOFFSET]_ZLIT0[4],          ; LOAD FPDOFFSET IN CASE PSL<FPD>=1
                                ;40407          INTPEND OR TIMER?,              ;
U 1622, 0586,CC37,0AF0,2047,04F7,0 ;40408       PUSH,NEXT/IE.SERV.IP.TS2        ; SERVICE PENDING INTERRUPT OR TIMER.
                                ;40409
                                ;40410          ;011--------------------------; CATCH RETURN+1 FROM IE.SERV.IP.TS,
U 1623, 0480,0036,4030,0047,0165,E ;40411       NEXT/MM.GET.BUT.PTE05           ; TIMER WAS SERVICED CONTINUE WITH MISS
                                ;40412
                                ;40413          ;100--------------------------; NO ACCESS TRYING TO FETCH PTE, TAKE AN
U 1624, 0480,0036,4030,0047,00F4,E ;40414       NEXT/IE.ACV.RBACK               ; ACV AND ROLL BACK REGARDLESS OF FPD
                                ;40415
                                ;40416          ;101--------------------------; NOT VALID TRYING TO FETCH PTE, TAKE A
U 1625, 0480,0036,4030,0047,00FB,1 ;40417       NEXT/IE.TNV.RBACK               ; TNV AND ROLL BACK REGARDLESS OF FPD.
                                ;40418
                                ;40419          ;110--------------------------; SUCCESSFULLY FETCHED THE NEW PTE.
                                ;40420          R[MM.TEMP5]_M[MDR].RL.9,        ; THE NEW PTE IS IN MDR,
                                ;40421          PTE CHECK READ?,                ; CHECK IT FOR READ ACCESS.
U 1626, 0C85,2477,07B3,8056,0963,4 479* ;40422  NEXT/MM.GET.BUT.PTE20           ;
                                ;40423
                                ;40424          ;111--------------------------; SUCCESSFULLY FETCHED THE NEW PTE.
                                ;40425          R[MM.TEMP5]_M[MDR].RL.9,        ; THE NEW PTE IS IN MDR,
                                ;40426          PTE CHECK READ?,                ; CHECK IT FOR READ ACCESS.
U 1627, 0C85,2477,07B3,8056,0963,4 479* ;40427  NEXT/MM.GET.BUT.PTE20           ;
                                ;40428 =100
                                ;40429 MM.GET.BUT.PTE20:
                                ;40430          ;100--------------------------; NO READ ACCESS,
                                ;40431          M[ERRCOD]_ZLIT0[0],             ; SET ERROR CODE AND TAKE AN
U 1634, 0986,BC37,0030,0047,00F4,E ;40432       NEXT/IE.ACV.RBACK               ; ACV AND ROLL BACK REGARDLESS OF FPD.
                                ;40433
                                ;40434          ;101--------------------------; READ ACCESS OK BUT NOT VALID,
                                ;40435          M[ERRCOD]_ZLIT0[0],             ; SET ERROR CODE AND TAKE A
U 1635, 0986,BC37,0030,0047,00FB,1 ;40436       NEXT/IE.TNV.RBACK               ; TNV AND ROLL BACK REGARDLESS OF FPD.
                                ;40437
                                ;40438 =111     ;111--------------------------;
U 1637, 0480,05BE,4039,C4A7,0178,4 ;40439       VA_R[MM.TEMP3]                  ; LOAD FAULTING ADDRESS INTO VA.
```

```
                                    ;40440      ;--------------------------------;
U 1784, 0081,B5BE,4033,8507,0178,5  ;40441      TB_R[MM.TEMP5]                   ; LOAD THE NEW PTE INTO THE TB.
                                    ;40442
                                    ;40443      ;--------------------------------;
                                    ;40444      MDR_R[MM.TEMP2],                 ;
U 1785, 0880,05BE,4039,8467,0166,F  ;40445      NEXT/MM.GET.BUT.FLUSH            ;
```

```
                                ;40446 .TOC "  Memory Management                : MM.GET.PTE"
                                ;40447
                                ;40448 ;*****************************************************************************
                                ;40449 ;       Entry points    MM.GET.PTE
                                ;40450 ;
                                ;40451 ;       Input           VA              Faulting address
                                ;40452 ;
                                ;40453 ;       Output          MDR             PTE
                                ;40454 ;
                                ;40455 ;       Resources       ERRCOD          Error code if TNV or ACV
                                ;40456 ;                       MM.TEMP0        Save PSL (double miss only)
                                ;40457 ;                       MM.TEMP5        Save VAPTE (double miss only)
                                ;40458 ;                       VA
                                ;40459 ;
                                ;40460 ;       MM.GET.PTE FETCHES THE PTE FOR THE VIRTUAL ADDRESS IN VA.
                                ;40461 ;       IF THERE IS A LENGTH VIOLATION OR AN ACV OR TNV FETCHING THE
                                ;40462 ;       SPTE FOR A PROCESS PTE THE PROPER BITS ARE SET IN THE ERROR
                                ;40463 ;       CODE IN ERRCOD AND SEPERATE RETURNS ARE TAKEN FOR ACV AND
                                ;40464 ;       TNV.  IF THE VA CONTAINS A SYSTEM ADDRESS THE PTE IS LOADED INTO
                                ;40465 ;       MDR AND A RETURN TAKEN INDICATING THAT THE VA NOW CONTAINS
                                ;40466 ;       THE PHYSICAL ADDRESS OF THE SPTE IN MDR.  IF THE VA CONTAINS A
                                ;40467 ;       PROCESS SPACE ADDRESS THE PTE IS LOADED INTO MDR AND A RETURN TAKEN
                                ;40468 ;       INDICATING THAT THE VA NOW CONTAINS THE VIRTUAL ADDRESS OF THE
                                ;40469 ;       PTE IN MDR.
                                ;40470 ;
                                ;40471 ;       CHARLIE'S FLOWS 1.3
                                ;40472 ;*****************************************************************************
                                ;40473 ;       TB PARITY ERROR RECOVERY
                                ;40474 ;
                                ;40475 ;       Resources       FPDOFFSET       bit<29> as a flag of MM.GET.WRITE.PTE
                                ;40476 ;                                       bus cycle in progress
                                ;40477 ;*****************************************************************************
                                ;40478
                                ;40479 15D8:   ;*********FORCE ADDRESS*********;
                                ;40480 MM.GET.PTE:
                                ;40481         ;00-----------------------------; VA IS IN P0 SPACE.
                                ;40482         WB_M[VA].VPN-R[P0LR],           ; DO LENGTH CHECK.
                                ;40483         SIGND CMP?,SIZE[LONG],          ;
U 15D8, 0481,B37F,0B6A,4047,095D,1 407*         ;40484         NEXT/MM.GET.PTE.P0              ;
                                ;40485
                                ;40486 15D9:   ;01-----------------------------; VA IS IN P1 SPACE.
                                ;40487         WB_M[VA].VPN-R[P1LR],           ; DO LENGTH CHECK.
                                ;40488         SIGND CMP?,SIZE[LONG],          ;
U 15D9, 0081,B37F,0B6A,C047,095E,1 407*         ;40489         NEXT/MM.GET.PTE.P1              ;
                                ;40490
                                ;40491 15DA:   ;10-----------------------------; VA IS IN SYSTEM SPACE.
                                ;40492         WB_M[VA].VPN-R[SLR]             ; DO LENGTH CHECK.
                                ;40493         SIGND CMP?,SIZE[LONG],          ;
U 15DA, 0881,B37F,0B6B,4047,095C,1 407*         ;40494         NEXT/MM.GET.PTE.SYS             ;
                                ;40495
                                ;40496 15DB:   ;11-----------------------------; VA IS IN RESERVED SPACE.
                                ;40497         M[ERRCOD] ?LIT0[1],             ; SET ERROR CODE AND
U 15DB, 0186,BC37,00B0,0847,0000,4              ;40498         RETURN [+4]                     ; TAKE ACV RETURN.
```

```
                                        ;40499 =01
                                        ;40500 MM.GET.PTE.SYS:
                                        ;40501          ;01------------------------------;
                                        ;40502          M[ERRCOD]_ZLIT0[1],              ; SET ERROR CODE AND
U 15C1, 0186,BC37,00B0,0847,0000,4      ;40503          RETURN [+4]                      ; TAKE ACV RETURN.
                                        ;40504
                                        ;40505          ;11------------------------------;
U 15C3, 0881,B33D,003B,04A7,0178,6      ;40506          VA_R[SBR]+M[VA].PTX              ; LOAD VA WITH THE PA OF THE PTE.
                                        ;40507
                                        ;40508          ;------------------------------;
U 1786, 0980,0CB7,0030,0687,0178,7      ;40509          MEMSCAR_ZLIT24[0]                ; LOAD ADDRESS OF MME BIT
                                        ;40510
                                        ;40511          ;------------------------------;
U 1787, 0880,05B7,0030,0607,0178,8      ;40512          MEMSCR_0                         ; TURN OFF MEMORY MANAGEMENT
                                        ;40513
                                        ;40514          ;------------------------------;
U 1788, 0181,BC13,8030,1CA7,0178,9      ;40515          VA_M[VA].ANDNOT.ZLIT0[3]         ; IGNORE LOW 2 BITS OF ADDRESS.
                                        ;40516
                                        ;40517          ;------------------------------; READ.PHY GRONKS 2ND MDR SO, MM IS
U 1789, 0C80,0036,4030,0042,0178,A      ;40518          READ.NOTRAP                      ; TURNED OFF TO DO A PHYSICAL READ.
                                        ;40519
                                        ;40520          ;------------------------------;
                                        ;40521          MEMSCR_-1,                       ; TURN MEMORY MANAGEMENT BACK ON.
U 178A, 0880,0E77,00B0,0607,0000,6      ;40522          RETURN [+6]                      ; MDR=PTE, VA=PAPTE.
                                        ;40523
                                        ;40524 =01
                                        ;40525 MM.GET.PTE.P0:
                                        ;40526          ;01------------------------------; VPN IS .GE. P0LR,
                                        ;40527          M[ERRCOD]_ZLIT0[1],              ; SET ERROR CODE AND
U 15D1, 0186,BC37,00B0,0847,0000,4      ;40528          RETURN [+4]                      ; TAKE ACV RETURN.
                                        ;40529
                                        ;40530          ;11------------------------------; VPN IS .LT. P0LR,
                                        ;40531          VA_R[P0BR]+M[VA].PTX,            ; LOAD VA WITH THE VA OF THE PTE.
U 15D3, 0C81,B33D,06FA,04A7,095F,8 371* ;40532          WB<31-30>?,NEXT/MM.GET.PTE.PX    ; CHECK THAT VAPTE IS IN SYSTEM SPACE.
                                        ;40533
                                        ;40534 =01
                                        ;40535 MM.GET.PTE.P1:
                                        ;40536          ;01------------------------------; VPN IS .GE. P1LR,
                                        ;40537          VA_R[P1BR]+M[VA].PTX,            ; LOAD VA WITH THE PA OF THE PTE.
U 15E1, 0881,B33D,06FA,84A7,095F,8 371* ;40538          WB<31-30>?,NEXT/MM.GET.PTE.PX    ; CHECK THAT VAPTE IS IN SYSTEM SPACE.
                                        ;40539
                                        ;40540          ;11------------------------------; VPN IS .LT. P1LR,
                                        ;40541          M[ERRCOD]_ZLIT0[1],              ; SET ERROR CODE AND
U 15E3, 0186,BC37,00B0,0847,0000,4      ;40542          RETURN [+4]                      ; TAKE ACV RETURN.
                                        ;40543
                                        ;40544 =00
                                        ;40545 MM.GET.PTE.PX:
                                        ;40546          ;00------------------------------; VAPTE IS IN P0 SPACE,
                                        ;40547          M[ERRCOD]_ZLIT0[3],              ; SET ERROR CODE AND
U 15F8, 0586,BC37,00B0,1847,0000,4      ;40548          RETURN [+4]                      ; TAKE ACV RETURN.
                                        ;40549
                                        ;40550          ;01------------------------------; VAPTE IS IN P1 SPACE,
                                        ;40551          M[ERRCOD]_ZLIT0[3],              ; SET ERROR CODE AND
U 15F9, 0586,BC37,00B0,1847,0000,4      ;40552          RETURN [+4]                      ; TAKE ACV RETURN.
```

```
                                ;40553          ;10*****************************;
                                ;40554          PROBE READ MODE ON WB?,         ; PROBE VAPTE TO AVOID NESTED TBMISS.
                                ;40555          SIZE[BYTE],WB_R[ZERO],          ; PROBE MODE = KERNAL.
                                ;40556          PATCH,                          ; ** jump to PCS **
U 15FA, 1480,05BE,478D,805E,0967,C 479*         ;40557          NEXT/MM.GET.PTE.PX20            ;
                                ;40558
                                ;40559          ;11-----------------------------; VAPTE IS IN RESERVED SPACE,
                                ;40560          M[ERRCOD]_ZLIT0[3],             ; SET ERROR CODE AND
U 15FB, 0586,BC37,00B0,1847,0000,4              ;40561          RETURN [+4]                     ; TAKE ACV RETURN.
                                ;40562
                                ;40563 =1100
                                ;40564 MM.GET.PTE.PX20:
                                ;40565          ;1100############################; PTE FOR VAPTE NOT IN TB.
                                ;40566          R[MM.TEMP5]_M[VA],              ; SAVE VAPTE.
                                ;40567          FORCE 32 BITS OF VA,            ; BUS/PRB.RD FORCES ALL 32 IF IN C MODE.
U 167C, 0085,B592,4033,805F,0178,D              ;40568          NEXT/MM.GET.PTE.PX32            ; ** code replaced in PCS **
                                ;40569
                                ;40570 =1110    ;1110############################;
                                ;40571          M[ERRCOD]_ZLIT0[1],             ; SET ERROR CODE AND
U 167E, 0186,BC37,00B0,0847,0000,4              ;40572          RETURN [+4]                     ; TAKE ACV RETURN.
                                ;40573
                                ;40574 MM.GET.PTE.PX25:
                                ;40575          ;1111############################;
                                ;40576          VA_M[VA].ANDNOT.ZLIT0[3],       ; FORCE VAPTE TO BE ALLIGNED.
U 167F, 0981,BC13,8030,1CBF,0178,B              ;40577          FORCE 32 BITS OF VA             ; BUS/PRB.RD FORCES ALL 32 IF IN C MODE.
                                ;40578
                                ;40579          ;############################;
                                ;40580          READ.NOTRAP,                    ; FETCH THE PTE.
U 178B, 0C80,0036,4030,0042,0178,C              ;40581          NEXT/MM.GET.PTE.PX25.1
                                ;40582
                                ;40583 2A7C:    ;1100=========================;
                                ;40584          R[MM.TEMP5]_M[VA],              ; SAVE VAPTE.
                                ;40585          FORCE 32 BITS OF VA,            ; BUS/PRB.RD FORCES ALL 32 IF IN C MODE.
U 2A7C, 0085,B592,4033,805F,0178,D              ;40586          NEXT/MM.GET.PTE.PX32            ; ** code replaced in PCS **
                                ;40587
                                ;40588 2A7E:    ;1110=========================; ** code in PCS **
                                ;40589          M[ERRCOD]_ZLIT0[2],             ; SET ERRCOD AND
U 2A7E, 0186,BC37,00B0,1047,0000,4              ;40590          RETURN [+4]                     ; TAKE ACV RETURN.
                                ;40591
                                ;40592 2A7F:    ;1111=========================;
                                ;40593          VA_M[VA].ANDNOT.ZLIT0[3],       ; FORCE VAPTE TO BE ALLIGNED.
U 2A7F, 0181,BC13,8030,1CBF,02A7,B              ;40594          FORCE 32 BITS OF VA             ; BUS/PRB.RD FORCES ALL 32 IF IN C MODE.
                                ;40595
                                ;40596 2A7B:    ;=============================;
U 2A7B, 0586,CC52,4030,1047,02A7,D              ;40597          M[FPDOFFSET]_MB.OR.ZLIT28[02]   ; TELL TB RECOVERY THIS IS PTE FETCH
                                ;40598
                                ;40599 2A7D:    ;=============================;
U 2A7D, 0480,0036,4030,0042,02A7,A              ;40600          READ.NOTRAP                     ; FETCH THE PTE.
                                ;40601                                          ; If TB parity error traped here, takes
                                ;40602                                          ; RETURN [-1] in recovery routine.
                                ;40603
                                ;40604 2A7A:    ;=============================;
U 2A7A, 0186,CC53,8030,1047,0178,C              ;40605          M[FPDOFFSET]_MB.ANDNOT.ZLIT28[2]; CLEAR PTE FETCH FLAG
                                ;40606
                                ;40607 MM.GET.PTE.PX25.1:
```

```
                                    ;40608          ;--------------------------------;
                                    ;40609          R[MM.TEMP5]_M[MDR].RL.9,        ; SAVE ROTATED PTE
1780, 0885,2477,0083,8047,0000,7    ;40610          RETURN [+7]                     ; TAKE GOOD RETURN.
```

```
                                ;40611          ;*****************************************************************************
                                ;40612          ;
                                ;40613          ; THE NEXT SECTION IS FOR PRCOESS SPACE TB MISSES THAT GET A TB MISS
                                ;40614          ; TRYING TO FETCH THE PROCESS SPACE PTE FROM VIRTUAL SYSTEM SPACE.
                                ;40615          ;
                                ;40616          ;*****************************************************************************
                                ;40617
                                ;40618 MM.GET.PTE.PX32:
                                ;40619          ;-------------------------------;
                                ;40620          FORCE 32 BITS OF VA,            ; BUS/PRB.RD FORCES ALL 32 IF IN C MODE.
                                ;40621          WB_M[VA].VPN-R[SLR],            ; PROCESS SYSTEM SPACE TBMISS.
U 178D, 0081,B37F,0B6B,405F,0960,9 407*         ;40622          SIGND CMP?,SIZE[LONG]           ; DO LENGTH CHECK.
                                ;40623
                                ;40624 =01      ;01-------------------------------;
                                ;40625          M[ERRCOD]_#LITO[3],             ; SET ERROR CODE AND
U 1609, 0586,BC37,00B0,1847,0000,4              ;40626          RETURN [+4]                     ; TAKE ACV RETURN.
                                ;40627
                                ;40628          ;11-------------------------------;
                                ;40629          FORCE 32 BITS OF VA,            ; BUS/PRB.RD FORCES ALL 32 IF IN C MODE.
U 160B, 0081,B33D,003B,04BF,0178,E              ;40630          VA_R[SBR]+M[VA].PTX             ; LOAD VA WITH THE PA OF THE PTE.
                                ;40631
                                ;40632          ;-------------------------------; FETCH THE PER PROCESS SPACE
U 178E, 0480,0036,4030,0040,0178,F              ;40633          READ.PHY                        ; PAGE TABLE PTE FROM SYSTEM SPACE.
                                ;40634
                                ;40635          ;-------------------------------;
U 178F, 0080,058E,4033,84A7,0179,0              ;40636          VA_R[MM.TEMP5]                  ; RESTORE VAPTE
                                ;40637
                                ;40638          ;-------------------------------;
U 1790, 0C85,2477,0033,8047,0179,1              ;40639          R[MM.TEMP5]_MLMDR].RL.9         ; SAVE ROTATED SPTE
                                ;40640
                                ;40641          ;-------------------------------; PROBE THE PAGE TABLE PTE
                                ;40642          R[MM.TEMP5]_M[MDR].RL.9,        ; THE NEW SPTE IS IN MDR,
U 1791, 0C85,2477,07B3,8057,0969,C 479*         ;40643          PTE CHECK READ KERNAL?          ;
                                ;40644
                                ;40645 =1100    ;1100-----------------------------; NO READ ACCESS TO PROCESS PAGE TABLE.
                                ;40646          M[ERRCOD]_#LITO[2],             ; SET ERROR CODE AND
U 169C, 0186,BC37,0080,1047,0000,4              ;40647          RETURN [+4]                     ; TAKE ACV RETURN.
                                ;40648
                                ;40649          ;1101-----------------------------; NOT VALID PROCESS PAGE TABLE.
                                ;40650          M[ERRCOD]_#LITO[2],             ; SET ERROR CODE AND
U 169D, 0986,BC37,0080,1047,0000,5              ;40651          RETURN [+5]                     ; TAKE TNV RETURN.
                                ;40652
                                ;40653 =1111    ;1111-------------------------------;
                                ;40654          FORCE 32 BITS OF VA,            ; BUS/PRB.RD FORCES ALL 32 IF IN C MODE.
                                ;40655          TB_R[MM.TEMP5],                 ; LOAD THE SYSTEM PTE INTO THE TB.
                                ;40656          PATCH,                          ; ** jump to PCS **
U 169F, 1881,B58E,4033,851F,0167,F              ;40657          NEXT/MM.GET.PTE.PX25            ;
```

```
                        ;40658 .TOC "  Memory Management                    : Subroutines for Probeing"
                        ;40659 .TOC "  Memory Management                    :       MM.PRB.WRITE.SIZ"
                        ;40660
                        ;40661 ;*****************************************************************************
                        ;40662 ;       Entry points    MM.PRB.WRITE.SIZ
                        ;40663 ;
                        ;40664 ;       Input           VA              Virtual address to be probed
                        ;40665 ;                       DSIZE signal    Size to be probed
                        ;40666 ;
                        ;40667 ;       Resources       MM.TEMP1        Save VA when calling MM.GET.WRITE.PTE
                        ;40668 ;                       MM.TEMP3        Save faulting address if ACV or TNV
                        ;40669 ;
                        ;40670 ;       Subroutines     IE.TNV
                        ;40671 ;                       IE.ACV
                        ;40672 ;                       MM.GET.WRITE.PTE
                        ;40673 ;                       MM.PRB.WRITE.NR
                        ;40674 ;
                        ;40675 ;       PRB.WRITE.SIZ PROBES THE ADDRESS IN VA FOR WRITE ACCESS USING THE
                        ;40676 ;       LENGTH SPECIFIED IN DSIZE.  IF THE ACCESS IS OK IT RETRJNS + 1
                        ;40677 ;       IF THERE IS AN ACV OR TNV THE PROPER FAULT IS INITIATED AND NO
                        ;40678 ;       RETURN IS TAKEN.
                        ;40679 ;
                        ;40680 ;       CHARLIE'S FLOWS 5.1
                        ;40681 ;*****************************************************************************
                        ;40682
                        ;40683
                        ;40684 1796:   ;*********FORCE ADDRESS*********;
                        ;40685 MM.PRB.WRITE.SIZ.00:
                        ;40686         ;-------------------------------;
U 1796, 0C80,0036,47B0,005D,0968,4 479* ;40687         PROBE WRITE?,SIZE[IDEP]         ; PROBE INITIAL VA.
                        ;40688
                        ;40689 1684:   ;*********FORCE ADDRESS*********;
                        ;40690 MM.PRB.WRITE.SIZ:
                        ;40691         ;0100---------------------------; PROBE NOT VALID (PTE NOT IN TB)
                        ;40692         R[MM.TEMP1]_M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 1684, 0485,B592,4033,C047,0561,8      ;40693         PUSH,NEXT/MM.PRB.WRT.TBM        ; LOAD THE PTE BACK IN THE TB.
                        ;40694
                        ;40695 1685:   ;0101---------------------------; CATCH RETURN+1 FROM MM.GET.WRITE.PTE
U 1685, 0C80,0036,4030,0047,00F6,1      ;40696         NEXT/IE.TNV                     ; TAKE TNV FAULT.
                        ;40697
                        ;40698 1686:
                        ;40699 MM_PCALL_ACV_ENTRY:
                        ;40700         ;0110---------------------------; PROBE NO ACCESS OR RETURN+2 FROM
                        ;40701         R[MM.TEMP3]_M[VA],              ; MM.GET.WRITE.PTE SAVE FAULTING
U 1686, 0C85,B592,4039,C047,00F5,D      ;40702         NEXT/IE.ACV                     ; ADDRESS AND TAKE ACV FAULT.
                        ;40703
                        ;40704 1687:   ;0111---------------------------; PROBE UNALIGNED BUT ACCESS OK OR
                        ;40705         R[MM.TEMP1]_M[VA]-1,            ; RETURN+3 FROM MM.GET.WRITE.PTE,
U 1687, 0485,BE51,0033,C047,0160,4      ;40706         NEXT/MM.PRB.WRITE.SIZ.10        ; PROBE VA+SIZE-1.
                        ;40707
                        ;40708 168F:   ;1111---------------------------;
U 168F, 0880,0036,40B0,0047,0000,1      ;40709         RETURN [+1]                     ; SIZE DEPENDENT ACCESS IS OK.
```

```
                                ;40710 =0
                                ;40711 MM.PRB.WRITE.SIZ.10:
                                ;40712          ;0-------------------------------;
                                ;40713          VA_R[MM.TEMP1]_RB+CONX.SIZ,     ;
                                ;40714          SIZE[IDEP],                     ;
U 1604, 0C84,073D,0033,C4A7,0579,A ;40715       PUSH,NEXT/MM.PRB.WRITE.NR       ; PROBE LAST BYTE.
                                ;40716
                                ;40717          ;1-------------------------------;
                                ;40718          R[MM.TEMP1]_RB-CONX.SIZ,        ;
U 1605, 0884,073C,0033,C047,0179,2 ;40719       SIZE[IDEP]                      ;
                                ;40720
                                ;40721          ;-------------------------------;
                                ;40722          VA_R[MM.TEMP1]+1,               ;
U 1792, 0C80,0E7C,00B3,C4A7,0000,1 ;40723       RETURN [+1]                     ; SIZE DEPENDENT ACCESS IS OK.
```

```
                                ;40724 .TOC "  Memory Management                :       PRB.WRITE.LONG"
                                ;40725
                                ;40726 ;*****************************************************************************
                                ;40727 ;       Entry points    MM.PRB.WRITE.LONG
                                ;40728 ;
                                ;40729 ;       Input           VA              Virtual address to be probed
                                ;40730 ;                       DSIZE signal    *********Must indicate byte***********
                                ;40731 ;
                                ;40732 ;       Resources       MM.TEMP1        Save VA when calling MM.PRB.WRT.TBM
                                ;40733 ;                       MM.TEMP3        Save faulting address if ACV or TNV
                                ;40734 ;
                                ;40735 ;       Subroutines     IE.TNV
                                ;40736 ;                       IE.ACV
                                ;40737 ;                       MM.PRB.WRT.TBM
                                ;40738 ;                       MM.PRB.WRITE.NR
                                ;40739 ;
                                ;40740 ;       MM.PRB.WRITE.LONG probes the longword specified by VA for write access.
                                ;40741 ;       It does a return+1 if the access is ok and takes a fault otherwise.
                                ;40742 ;       (Used only in queue instructions. Could be eliminated by slowing
                                ;40743 ;       down INSQUE and REMQUE, by 6 and 3 cycles respectively.)
                                ;40744 ;
                                ;40745 ;       CHARLIE'S FLOWS 1.7
                                ;40746 ;*****************************************************************************
                                ;40747
                                ;40748 1798:   ;*********FORCE ADDRESS*********;
                                ;40749 MM.PRB.WRITE.LONG:
                                ;40750         ;-------------------------------;
U 1798, 0080,0036,47A0,005D,096A,4 479*  ;40751         PROBE WRITE?,SIZE[LONG]         ; PROBE INITIAL VA.
                                ;40752
                                ;40753 =0100
                                ;40754         ;0100-----------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                ;40755         R[MM.TEMP1]_M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 16A4, 0485,B592,4033,C047,0561,8       ;40756         PUSH,NEXT/MM.PRB.WRT.TBM        ; LOAD THE PTE BACK IN THE TB.
                                ;40757
                                ;40758         ;0101-----------------------------; CATCH RETURN+1 FROM MM.PRB.WRT.TBM
U 16A5, 0C80,0036,4030,0047,00F6,1       ;40759         NEXT/IE.TNV                     ; TAKE TNV FAULT.
                                ;40760
                                ;40761         ;0110-----------------------------; PROBE NO ACCESS OR RETURN+2 FROM
                                ;40762         R[MM.TEMP3]_M[VA],              ; MM.PRB.WRT.TBM SAVE FAULTING
U 16A6, 0C85,B592,4039,C047,00F5,D       ;40763         NEXT/IE.ACV                     ; ADDRESS AND TAKE ACV FAULT.
                                ;40764
                                ;40765         ;0111-----------------------------; PROBE UNALIGNED BUT ACCESS OK OR
                                ;40766         R[MM.TEMP1]_M[VA]-1,            ; RETURN+3 FROM MM.PRB.WRT.TBM,
U 16A7, 0C85,BE51,0033,C047,0163,C       ;40767         NEXT/MM.PRB.WRITE.LONG.10       ; PROBE VA+SIZE-1.
                                ;40768
                                ;40769 =1111   ;1111-----------------------------;
U 16AF, 0880,0036,40B0,0047,0000,1       ;40770         RETURN [+1]                     ; SIZE DEPENDENT ACCESS IS OK.
```

```
                                    ;40771 =0
                                    ;40772 MM.PRB.WRITE.LONG.10:
                                    ;40773          ;0------------------------------;
                                    ;40774          VA_R[MM.TEMP1]_RB+CONX(4),      ;
U 163C, 0884,073D,0023,C4A7,0579,A  ;40775          PUSH,NEXT/MM.PRB.WRITE.NR       ; PROBE LAST BYTE.
                                    ;40776
                                    ;40777          ;1------------------------------;
U 163D, 0484,073C,0023,C047,0179,3  ;40778          R[MM.TEMP1]_RB-CONX(4)          ;
                                    ;40779
                                    ;40780          ;------------------------------;
                                    ;40781          VA_R[MM.TEMP1]+1,               ;
U 1793, 0C80,0E7C,00B3,C4A7,0000,1  ;40782          RETURN [+1]                     ; SIZE DEPENDENT ACCESS IS OK.
```

```
                                ;40783 .TOC "  Memory Management                :       PRB.WRITE.NR"
                                ;40784
                                ;40785 ;*****************************************************************************
                                ;40786 ;       Entry points    MM.PRB.WRITE.NR
                                ;40787 ;
                                ;40788 ;       Input           VA              Virtual address to be probed
                                ;40789 ;
                                ;40790 ;       Resources       MM.TEMP1        Save VA when calling MM.PRB.WRT.TBM
                                ;40791 ;                       MM.TEMP3        Save faulting address if ACV or TNV
                                ;40792 ;
                                ;40793 ;       Subroutines     IE.TNV
                                ;40794 ;                       IE.ACV
                                ;40795 ;                       MM.GET.WRITE
                                ;40796 ;
                                ;40797 ;       PRB.WRITE.NR IF BRANCHED TO WHILE DOING A BRANCH ON WB<1:0>=0? WILL
                                ;40798 ;       TAKE A RESERVED OPERAND FAULT IF THE ADDRESS ON WBUS IS NOT
                                ;40799 ;       LONGWORD ALIGNED,
                                ;40800 ;       OTHERWISE IT PROBES THE BYTE SPECIFIED BY VA FOR WRITE ACCESS.
                                ;40801 ;       IF THE ACCESS IS OK IT DOES A RETURN+1, IF THERE IS AN ACV OR TNV
                                ;40802 ;       THEN THE PROPER FAULT IS INITIATED.
                                ;40803 ;
                                ;40804 ;       CHARLIE'S FLOWS 5.1
                                ;40805 ;*****************************************************************************
                                ;40806
                                ;40807 179A:   ;*********FORCE ADDRESS*********;
                                ;40808 MM.PRB.WRITE.NR:
                                ;40809         ;-------------------------------;
U 179A, 0480,0036,4780,005D,096B,C 479*         ;40810         PROBE WRITE?,SIZE[BYTE]         ; PROBE VA.
                                ;40811
                                ;40812 =1100
                                ;40813         ;1100-----------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                ;40814         R[MM.TEMP1]_M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 16BC, 0485,B592,4033,C047,0561,8      ;40815         PUSH,NEXT/MM.PRB.WRT.TBM        ; LOAD THE PTE BACK IN THE TB.
                                ;40816
                                ;40817         ;1101-----------------------------; CATCH RETURN+1 FROM MM.PRB.WRT.TBM
U 16BD, 0C80,0036,4030,0047,00F6,1      ;40818         NEXT/IE.TNV                     ; TAKE A TNV FAULT.
                                ;40819
                                ;40820         ;1110-----------------------------; PROBE NO ACCESS OR RETURN+2 FROM
                                ;40821         R[MM.TEMP3]_M[VA],              ; MM.PRB.WRT.TBM,  SAVE FAULTING
U 16BE, 0C85,B592,4039,C047,00F5,D      ;40822         NEXT/IE.ACV                     ; ADDRESS AND TAKE AN ACV FAULT.
                                ;40823
                                ;40824         ;1111-----------------------------; WRITE ACCESS OK OR RETURN+3 FROM
U 16BF, 0880,0036,40B0,0047,0000,1      ;40825         RETURN [+1]                     ; MM.PRB.WRT.TBM, TAKE OK RETURN.
```

```
                                ;40826 .TOC "  Memory Management                :       PRB.WRITE1"
                                ;40827
                                ;40828 ;*************************************************************************
                                ;40829 ;       Entry points    MM.PRB.WRITE1
                                ;40830 ;
                                ;40831 ;       Input           VA              Virtual address to be probed
                                ;40832 ;                       TEMP2           Probe mode
                                ;40833 ;
                                ;40834 ;       Resources       MM.TEMP1        Save VA when calling MM.PRB.WRT.TBM
                                ;40835 ;                       MM.TEMP3        Save faulting address if ACV or TNV
                                ;40836 ;                       TEMP1           Save PSL if TB miss
                                ;40837 ;                       MDR             ******* CONTENTS DESTROYED **********
                                ;40838 ;
                                ;40839 ;       Subroutines     IE.TNV
                                ;40840 ;                       IE.ACV
                                ;40841 ;                       MM.GET.WRITE
                                ;40842 ;
                                ;40843 ;       PRB.WRITE1 PROBES THE BYTE SPECIFIED BY VA AND DSIZE
                                ;40844 ;       FOR WRITE ACCESS USING THE MODE SPECIFIED IN TEMP2.
                                ;40845 ;
                                ;40846 ;       CHARLIE'S FLOWS 5.1.X
                                ;40847 ;*************************************************************************
                                ;40848
                                ;40849 179B:   ;*********FORCE ADDRESS*********;
                                ;40850 MM.PRB.WRITE1:
                                ;40851         ;-----------------------------;
                                ;40852         PROBE WRITE MODE ON WB?,      ; PROBE INITIAL VA.
U 179B, 0480,2592,4780,005C,096C,C 479*  ;40853         SIZE[BYTE],                   ;
                                ;40854         WB_M[TEMP2]                   ;
                                ;40855
                                ;40856 =1100   ;1100-------------------------; PROBE NOT VALID (PTE NOT IN TB)
                                ;40857         M[TEMP1]_PSL,                 ; SAVE PSL AND PROCESS A TB MISS.
U 16CC, 0886,1036,4030,0087,0179,4       ;40858         NEXT/MM.PRB.WRITE1.10         ;
                                ;40859 =1110
                                ;40860 MM.PRB.WRITE1.05:
                                ;40861         ;1110-------------------------; PROBE NO ACCESS SAVE FAULTING
                                ;40862         R[MM.TEMP3]_M[VA],            ;
U 16CE, 0C85,B592,4039,C047,00F5,D       ;40863         NEXT/IE.ACV                   ; ADDRESS AND TAKE AN ACV FAULT.
                                ;40864
                                ;40865         ;1111-------------------------; ACCESS OK
U 16CF, 0880,0036,40B0,0047,0000,1       ;40866         RETURN [+1]                   ;
```

```
                                ;40867 MM.PRB.WRITE1.10:
                                ;40868          ;-------------------------------;
                                ;40869          PSL(PREV_CURM ISCURM_R[TEMP2]), ; CHANGE MODE FOR CHECKING ACCESS.
U 1794, 0060,05BE,4030,86A7,016D,C ;40870       SET MM.NOINT                    ; DISABLE INTERRUPTS TEMPORARILY
                                ;40871
                                ;40872 =1100    ;1100---------------------------;
                                ;40873          R[MM.TEMP1] M[VA],              ; SAVE VA AND PROCESS A TB MISS TO
U 16DC, 0485,B592,4033,C047,0561,8 ;40874       PUSH,NEXT/MM.PRB.WRT.TBM        ; LOAD THE PTE BACK IN THE TB.
                                ;40875
                                ;40876          ;1101---------------------------; CATCH RETURN+1 FROM MM.PRB.WRT.TBM
                                ;40877          PSL M[TEMP1],CLEAR MM.NOINT,    ; RESTORE ORIGINAL PSL, ALLOW INTERRUPTS
U 16DD, 0420,1002,403D,8007,00F6,1 ;40878       NEXT/IE.TNV                     ; TAKE A TNV FAULT.
                                ;40879
                                ;40880          ;1110---------------------------; NO ACCESS RETURN+2 FROM MM.PRB.WRT.TBM
                                ;40881          PSL M[TEMP1],CLEAR MM.NOINT,    ; RESTORE ORIGINAL PSL, ALLOW INTERRUPTS
U 16DE, 0C20,1002,403D,8007,016C,E ;40882       NEXT/MM.PRB.WRITE1.05           ; AND TAKE AN ACV FAULT.
                                ;40883
                                ;40884          ;1111---------------------------; ACCESS OK RETURN+3 FROM
                                ;40885          PSL M[TEMP1],CLEAR MM.NOINT,    ; RESTORE ORIGINAL PSL, ALLOW INTERRUPTS
U 16DF, 0020,1002,40BD,8007,0000,1 ;40886       RETURN [+1]                     ;
```

```
                                ;40887 .TOC "  Memory Management                 :       MM.PRB.WRT.TBM"
                                ;40888
                                ;40889 ;*************************************************************************
                                ;40890 ;       Entry points    MM.PRB.WRT.TBM
                                ;40891 ;
                                ;40892 ;       Subroutines     MM.GET.WRITE.PTE
                                ;40893 ;
                                ;40894 ;       This routine save MDR, processes a TB miss by calling
                                ;40895 ;       MM.GET.WRITE.PTE and restores MDR before returning.
                                ;40896 ;*************************************************************************
                                ;40897
                                ;40898 1618:   ;*********FORCE ADDRESS*********;
                                ;40899 MM.PRB.WRT.TBM:
                                ;40900         ;00-----------------------------;
                                ;40901         R[MM.TEMP2] M[MDR],             ; SAVE MDR AND FETCH THE PTE
U 1618, 0885,2592,4039,8047,055E,0 ;40902     PUSH,NEXT/MM.GET.WRITE.PTE      ;
                                ;40903
                                ;40904 1619:   ;01-----------------------------;
                                ;40905         MDR R[MM.TEMP2],                ; RESTORE MDR AND TAKE
U 1619, 0C80,05BE,40B9,8467,0000,1 ;40906     RETURN [+1]                     ; TNV RETURN
                                ;40907
                                ;40908 161A:   ;10-----------------------------;
                                ;40909         MDR R[MM.TEMP2],                ; RESTORE MDR AND TAKE
U 161A, 0C80,05BE,40B9,8467,0000,2 ;40910     RETURN [+2]                     ; ACV RETURN
                                ;40911
                                ;40912 161B:   ;11-----------------------------;
                                ;40913         MDR R[MM.TEMP2],                ; RESTORE MDR AND TAKE
U 161B, 0480,05BE,40B9,8467,0000,3 ;40914     RETURN [+3]                     ; ACCESS OK RETURN
```

```
                                ;40915 .TOC "  Memory Management                :       MM.PRB.READ.TBM"
                                ;40916
                                ;40917 ;*****************************************************************************
                                ;40918 ;       Entry points    MM.PRB.READ.TBM
                                ;40919 ;
                                ;40920 ;       Subroutines     MM.GET.READ.PTE
                                ;40921 ;
                                ;40922 ;       This routine save MDR, processes a TB miss by calling
                                ;40923 ;       MM.GET.READ.PTE and restores MDR before returning.
                                ;40924 ;*****************************************************************************
                                ;40925
                                ;40926 1628:   ;*********FORCE ADDRESS*********;
                                ;40927 MM.PRB.READ.TBM:
                                ;40928         ;00-----------------------------;
                                ;40929         R[MM.TEMP2] M[MDR],             ;
U 1628, 0085,2592,4039,8047,055C,0 ;40930      PUSH,NEXT/MM.GET.READ.PTE       ;
                                ;40931
                                ;40932 1629:   ;01-----------------------------;
                                ;40933         MDR R[MM.TEMP2],                ; RESTORE MDR AND TAKE
U 1629, 0C80,05BE,40B9,8467,0000,1 ;40934      RETURN [+1]                     ; TNV RETURN
                                ;40935
                                ;40936 162A:   ;10-----------------------------;
                                ;40937         MDR R[MM.TEMP2],                ; RESTORE MDR AND TAKE
U 162A, 0C80,05BE,40B9,8467,0000,2 ;40938      RETURN [+2]                     ; ACV RETURN
                                ;40939
                                ;40940 162B:   ;11-----------------------------;
                                ;40941         MDR R[MM.TEMP2],                ; RESTORE MDR AND TAKE
U 162B, 0480,05BE,40B9,8467,0000,3 ;40942      RETURN [+3]                     ; ACCESS OK RETURN
```

```
                                        ;40943 .TOC "CMODE.MIC"
                                        ;40944 .TOC "Revision 21.0     "
                                        ;40945 ;       Gerard Koeckhoven, Brian Allison
                                        ;40946
;40947  .NOBIN
;40948  .TOC "  Revision History"
;40949
;40950  ; REV   EXPLAINATION
;40951
;40952  ; 21    Cause ODD Address traps when the SP is odd in JSR, RTS, MFPx, MTPx and
;40953  ;       RTI/RTT
;40954  ; 20    Trap reserved opcodes that get confused with RTI, RTT, RTS, IOT and BPT.
;40955  ;       Fix ODD address check of operands with RTS,JSR, JMP
;40956  ;       Recover locations 147C and 147D
;40957  ; 19    Fix JSR to detect an illegal destination mode "0" at
;40958  ;       the beginning og the instruction.
;40959  ;       Fix the DIV to not set the "V" bit when the quotient equals 8000
;40960  ;       Fix the TSTB instruction to execute as a TSTB when the
;40961  ;       destination address is the PC.
;40962  ;       DIV:    The remainder is not of the same sign as the dividend
;40963  ;               when the operands are of a differend sign.
;40964  ; 18    DIV:    Zero is an acceptable "expected-negative" result.
;40965  ;               Fix problem that called it overflow and left result unwritten.
;40966  ;       MUL:    Fix bug mentioned in  17
;40967  ;       ASH:    Left shift should not overflow when result sign = original sign.
;40968  ;               Sign extending Q before the test fixes the problem.
;40969  ; 17    MUL; Vbit should not be set when result most neg no.
;40970  ;       Fix ASH to not set the V bit if the bits shifted out are of the
;40971  ;       same sign as the sign of the original and final operand.
;40972  ;16     Initial release.
                                        ;40973          .BIN
```

```
                                ;40974 .TOC   "       Compatibility Mode                : CLR(B)"
                                ;40975
                                ;40976 ;*****************************************************************************
                                ;40977 ;       CLR(B)  dest.wx
                                ;40978 ;          Input (dest is memory)       VA      Address of dest
                                ;40979 ;
                                ;40980 ;          Input (dest is register)     RNUM    # of register to clear
                                ;40981 ;*****************************************************************************
                                ;40982 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;40983 =1
                                ;40984 CM.CLR-REG:
                                ;40985         ;1-------------------------------;
U 0013, 0C80,0036,4BBC,C047,002C,C      ;40986         RNUM.EQ.7?                      ; TEST FOR CLR(B) PC
                                ;40987
                                ;40988 =00     ;00------------------------------; NOT REGISTER MODE PC
                                ;40989         R[DST.R].SIZ 0,WRITE NOTREG,    ; CLEAR MEM LOCATION OR REGISTER
                                ;40990         CCOP1,SIZE[IDEP],               ; SET THE CONDITION CODES
U 02CC, 0882,05B7,013C,4DDA,003F,9      ;40991         IRD1                            ;
                                ;40992
                                ;40993 =10     ;10------------------------------; REGISTER MODE PC
                                ;40994         PC R[ZERO],                     ; CLEAR ALL BITS IN PC
                                ;40995         CCOP1,SIZE[WORD],               ; SET CC'S
U 02CE, 0480,05BE,401D,8C87,0148,C      ;40996         NEXT/CM.NOP                     ; LET THE PC GET CAUGHT UP
                                ;40997
                                ;40998         ;11------------------------------; NOT REGISTER MODE PC
                                ;40999         R[DST.R].SIZ 0,WRITE NOTREG,    ; CLEAR MEM LOCATION OR REGISTER
                                ;41000         CCOP1,SIZE[IDEP],               ; SET THE CONDITION CODES
U 02CF, 0882,05B7,013C,4DDA,003F,9      ;41001         IRD1                            ;
                                ;41002
                                ;41003 CM.CLR-MEM:
                                ;41004         ;-------------------------------; NOT REGISTER MODE PC
                                ;41005         R[DST.R].SIZ 0,WRITE NOTREG,    ; CLEAR MEM LOCATION OR REGISTER
                                ;41006         CCOP1,SIZE[IDEP],               ; SET THE CONDITION CODES
U 03CA, 0482,0587,003C,4DDA,0072,2      ;41007         NEXT/CO.NOP                     ; WASTE A CYCLE
```

```
                        ;41008 .TOC     "       Compatibility Mode               : DEC(B)"
                        ;41009
                        ;41010 ;*************************************************************************
                        ;41011 ;       DEC(B)  dest.mx
                        ;41012 ;          Input (dest is memory)        VA      Address of dest
                        ;41013 ;                                        MDR     Contents of dest
                        ;41014 ;
                        ;41015 ;          Input (dest is register)      RNUM    # of register to dec
                        ;41016 ;*************************************************************************
                        ;41017 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;41018 =1
                        ;41019 CM.DEC-REG:
                        ;41020          ;-------------------------------;
U 0017, 0C80,05BF,488C,C467,0030,4      ;41021          MDR_GPR,R RNUM,EQ,7?            ; SEE IF SOME IDIOT WANTS TO DEC PC
                        ;41022
                        ;41023 =00      ;00-----------------------------; NOT REGISTER MODE PC
                        ;41024          R[DST.R].SIZ_M[MDR]-1,          ; DEC THE RIGHT REGISTER
                        ;41025          WRITE NOTREG,                   ; DO THE RIGHT THING IF MEMORY INST
                        ;41026          CCOP1,SIZE[IDEP],               ; SET THE CONDITION CODES
U 0304, 0883,2E51,013C,4DDA,003F,9      ;41027          IRD1            ;
                        ;41028
                        ;41029 =10      ;10-----------------------------; REGISTER MODE PC
                        ;41030          PC_M[PC]-ZLIT0[1],              ; SCREW UP THE PC
                        ;41031          CCOP1,SIZE[WORD],               ; SET THE CC'S JUST FOR THE HELL OF IT
U 0306, 0981,AC10,0010,0C87,0149,7      ;41032          NEXT/CM.TEST.PC.ODD             ; GO CAUSE AN ODD ADDRESS TRAP
                        ;41033
                        ;41034          ;11-----------------------------; NOT REGISTER MODE PC
                        ;41035          R[DST.R].SIZ_M[MDR]-1,          ; DEC THE RIGHT REGISTER
                        ;41036          WRITE NOTREG,                   ; DO THE RIGHT THING IF MEMORY INST
                        ;41037          CCOP1,SIZE[IDEP],               ; SET THE CONDITION CODES
U 0307, 0883,2E51,013C,4DDA,003F,9      ;41038          IRD1            ;
                        ;41039
                        ;41040 CM.DEC-MEM:
                        ;41041          ;-------------------------------; NOT REGISTER MODE PC
                        ;41042          R[DST.R].SIZ_M[MDR]-1,          ; DEC THE RIGHT REGISTER
                        ;41043          WRITE NOTREG,                   ; DO THE RIGHT THING IF MEMORY INST
                        ;41044          CCOP1,SIZE[IDEP],               ; SET THE CONDITION CODES
U 030B, 0483,2E51,003C,4DDA,0072,2      ;41045          NEXT/CO.NOP     ;
```

```
                        ;41046 .TOC     "       Compatibility Mode              : INC(B)"
                        ;41047
                        ;41048 ;*************************************************************************
                        ;41049 ;       INC(B)  dest.mx
                        ;41050 ;          Input (dest is memory)       VA      Address of dest
                        ;41051 ;                                       MDR     Contents of dest
                        ;41052 ;
                        ;41053 ;          Input (dest is register)     RNUM    # of register to inc
                        ;41054 ;*************************************************************************
                        ;41055 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;41056 =1
                        ;41057 CM.INC-REG:
                        ;41058         ;1-------------------------------;
U 001B, 0480,05BE,4BBC,C467,0030,C      ;41059         MDR_GPR.R RNUM.EQ.7?            ; SEE IF SOME IDIOT WANTS TO DEC PC
                        ;41060
                        ;41061 =00     ;00------------------------------; NOT REGISTER MODE PC
                        ;41062         R[DST.R].SIZ_M[MDR]+1,          ; ADD ONE TO THE CORRECT REGISTER
                        ;41063         WRITE NOTREG,                   ; TAKE CARE OF MEMORY DEST
                        ;41064         CCOP1,SIZE[IDEP],               ; SET CC'S
U 030C, 0C83,2E50,013C,4DDA,003F,9      ;41065         IRD1                            ;
                        ;41066
                        ;41067 =10     ;10------------------------------; REGISTER MODE PC
                        ;41068         PC_M[PC]+2,LIT0[1],             ; INC THE PC
                        ;41069         CCOP1,SIZE[WORD],               ; SET CC'S
U 030E, 0D81,AC11,0010,0C87,0149,7      ;41070         NEXT/CM.TEST.PC.ODD             ; CAUSE AN ODD ADDRESS TRAP
                        ;41071
                        ;41072         ;11------------------------------; NOT REGISTER MODE PC
                        ;41073         R[DST.R].SIZ_M[MDR]+1,          ; ADD ONE TO THE CORRECT REGISTER
                        ;41074         WRITE NOTREG,                   ; TAKE CARE OF MEMORY DEST
                        ;41075         CCOP1,SIZE[IDEP],               ; SET CC'S
U 030F, 0C83,2E50,013C,4DDA,003F,9      ;41076         IRD1                            ;
                        ;41077
                        ;41078 CM.INC-MEM:
                        ;41079         ;-------------------------------; NOT REGISTER MODE PC
                        ;41080         R[DST.R].SIZ_M[MDR]+1,          ; ADD ONE TO THE CORRECT REGISTER
                        ;41081         WRITE NOTREG,                   ; TAKE CARE OF MEMORY DEST
                        ;41082         CCOP1,SIZE[IDEP],               ; SET CC'S
U 030C, 0083,2E50,003C,4DDA,0072,2      ;41083         NEXT/CO.NOP                     ;
```

```
                                ;41084 .TOC     "       Compatibility Mode              : NEG(B)"
                                ;41085
                                ;41086 ;*************************************************************************
                                ;41087 ;       NEG(B)  dest.mx
                                ;41088 ;          Input (dest is memory)       VA      Address of dest
                                ;41089 ;                                       MDR     Contents of dest
                                ;41090 ;
                                ;41091 ;          Input (dest is register)     RNUM    # of register to negate
                                ;41092 ;*************************************************************************
                                ;41093 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41094 =1
                                ;41095 CM.NEG-REG:
                                ;41096         ;1-------------------------------;
U 0061, 0480,05BE,4BBC,C467,0031,4  ;41097         MDR_GPR.R RNUM.EQ.7?          ; SEE IF SOME IDIOT WANTS TO DEC PC
                                ;41098
                                ;41099 =00     ;00------------------------------; NOT REGISTER MODE PC
                                ;41100         R[DST.R].SIZ_-M[MDR],         ; NEGATE THE SPECIFIED REGISTER
                                ;41101         WRITE NOTREG,                 ; TAKE CARE OF MEMORY DEST
                                ;41102         CCOP2,SIZE[IDEP],             ; SET CC'S
U 0314, 0883,2593,013C,55DA,003F,9  ;41103         IRD1                          ;
                                ;41104
                                ;41105 =10     ;10------------------------------; REGISTER MODE PC
                                ;41106         PC_-M[PC],                    ; NEGATE PC
                                ;41107         CCOP2,SIZE[WORD],             ; SET CC'S
U 0316, 0481,A003,001D,9487,0148,C  ;41108         NEXT/CM.NOP                   ;
                                ;41109
                                ;41110         ;11------------------------------; NOT REGISTER MODE PC
                                ;41111         R[DST.R].SIZ_-M[MDR],         ; NEGATE THE SPECIFIED REGISTER
                                ;41112         WRITE NOTREG,                 ; TAKE CARE OF MEMORY DEST
                                ;41113         CCOP2,SIZE[IDEP],             ; SET CC'S
U 0317, 0883,2593,013C,55DA,003F,9  ;41114         IRD1                          ;
                                ;41115
                                ;41116 CM.NEG-MEM:
                                ;41117         ;-------------------------------; NOT REGISTER MODE PC
                                ;41118         R[DST.R].SIZ_-M[MDR],         ; NEGATE THE SPECIFIED REGISTER
                                ;41119         WRITE NOTREG,                 ; TAKE CARE OF MEMORY DEST
                                ;41120         CCOP2,SIZE[IDEP],             ; SET CC'S
U 03CD, 0483,2593,003C,55DA,0072,2  ;41121         NEXT/CO.NOP                   ;
```

```
                                ;41122 .TOC     "       Compatibility Mode               : TST(B)"
                                ;41123
                                ;41124 ;*****************************************************************************
                                ;41125 ;       TST(B)  dest.rx
                                ;41126 ;          Input (dest is memory)        VA      Address of dest
                                ;41127 ;                                        MDR     Contents of dest
                                ;41128 ;
                                ;41129 ;          Input (dest is register)      RNUM    # of register to test
                                ;41130 ;*****************************************************************************
                                ;41131 .RFGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41132 =1
                                ;41133 CM.TST-REG:
                                ;41134          ;1-------------------------------;
U 0067, 0C80,05BE,4BBC,C467,0032,C  ;41135          MDR_GPR.R RNUM.EQ.7?            ; SEE IF SOME IDIOT WANTS TO DEC PC
                                ;41136
                                ;41137 =00      ;00------------------------------; NOT REGISTER MODE PC
                                ;41138          WB_M[MDR],                      ; DRIVE THING TO TEST ONTO WBUS
                                ;41139          CCOP1,SIZE[IDEP],               ; SET CC'S
U 032C, 0081,2592,4130,0847,003F,9  ;41140          IRD1                            ;
                                ;41141
                                ;41142 =10      ;10------------------------------; REGISTER MODE PC
                                ;41143          WB_M[PC],                       ; TEST PC
                                ;41144          CCOP1,SIZE[IDEP],               ; SET CC'S
U 032E, 0481,A592,4130,0847,003F,9  ;41145          IRD1                            ;
                                ;41146
                                ;41147          ;11------------------------------; NOT REGISTER MODE PC
                                ;41148          WB_M[MDR],                      ; DRIVE THING TO TEST ONTO WBUS
                                ;41149          CCOP1,SIZE[IDEP],               ; SET CC'S
U 032F, 0081,2592,4130,0847,003F,9  ;41150          IRD1                            ;
                                ;41151
                                ;41152 CM.TST-MEM:
                                ;41153          ;-------------------------------; NOT REGISTER MODE PC
                                ;41154          WB_M[MDR],                      ; DRIVE THING TO TEST ONTO WBUS
                                ;41155          CCOP1,SIZE[IDEP],               ; SET CC'S
U 03CE, 0C81,2592,4030,0847,0072,2  ;41156          NEXT/CO.NOP                     ;
```

```
                                ;41157 .TOC     "       Compatibility Mode               : COM(B)"
                                ;41158
                                ;41159 ;*****************************************************************************
                                ;41160 ;       COM(B)  dest.mx
                                ;41161 ;          Input (dest is memory)        VA      Address of dest
                                ;41162 ;                                        MDR     Contents of dest
                                ;41163 ;
                                ;41164 ;          Input (dest is register)      RNUM    # of register to com
                                ;41165 ;*****************************************************************************
                                ;41166 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41167 =1
                                ;41168 CM.COM-REG:
                                ;41169          ;1-------------------------------;
                                ;41170          Q_0,                            ; TO USE IN SUBTRACT LATER
U 006B, 0C80,05B7,1BBC,C047,0033,4  ;41171      RNUM.EQ.7?                      ; SEE IF SOME IDIOT WANTS TO DEC PC
                                ;41172
                                ;41173 =00      ;00-------------------------------; NOT REGISTER MODE PC
                                ;41174          R[GPR.R],SIZ_Q-RB-1,            ; COMPLEMENT CONTENTS OF REGISTER
                                ;41175          CCOP2,SIZE[IDEP],               ; SET CC'S
U 0334, 0482,00BB,013C,D047,003F,9  ;41176      IRD1                            ;
                                ;41177
                                ;41178 =10      ;10-------------------------------; REGISTER MODE PC
                                ;41179          PC_Q-M[PC]-1,                   ; COMPLEMENT PC
                                ;41180          CCOP2,SIZE[WORD],               ; SET CC'S
U 0336, 0481,A08B,0010,1487,0149,7  ;41181      NEXT/CM.TEST.PC.ODD             ; CAUSE UDD ADDERSS TRAP
                                ;41182
                                ;41183          ;11-------------------------------; NOT REGISTER MODE PC
                                ;41184          R[GPR.R],SIZ_Q-RB-1,            ; COMPLEMENT CONTENTS OF REGISTER
                                ;41185          CCOP2,SIZE[IDEP],               ; SET CC'S
U 0337, 0482,00BB,013C,D047,003F,9  ;41186      IRD1                            ;
                                ;41187
                                ;41188
                                ;41189 CM.COM-MEM:
                                ;41190          ;-------------------------------;
                                ;41191          WRITE R[ZERO]-M[MDR]-1,         ; COMPLEMENT THE CONTENTS OF MDR
                                ;41192          CCOP2,SIZE[IDEP],               ; SET CC'S
U 03CF, 0881,2083,003D,95D8,0072,2  ;41193      NEXT/CO.NOP                     ; WASTE A CYCLE
```

```
                                ;41194 .TOC     ''      Compatibility Mode              : ASR(B)''
                                ;41195
                                ;41196 ;*************************************************************************
                                ;41197 ;       ASR(B)  dest.mx
                                ;41198 ;          Input (dest is memory)       VA      Address of dest
                                ;41199 ;                                       MDR     Contents of dest
                                ;41200 ;
                                ;41201 ;          Input (dest is register)     RNUM    # of register to shift
                                ;41202 ;*************************************************************************
                                ;41203 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41204 CM.ASR-MEM:
                                ;41205         ;--------------------------------;
U 03D0, 0C81,2816,403D,8467,0140,1  ;41206         MDR_SEXT(M[MDR]),SIZE[IDEP]    ; SIGN EXTEND TO GET RIGHT SHIFT IN
                                ;41207
                                ;41208 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41209         ;--------------------------------;
U 1401, 0580,0EF6,4030,0847,0140,2  ;41210         PL_[1]                         ; CONSTANT TO ROTATE BY
                                ;41211
                                ;41212         ;--------------------------------;
                                ;41213         WRITE M[MDR].RR.P,             ; SHIFT RIGHT BY ONE
                                ;41214         CCOP1,SIZE[IDEP],              ; SET CC'S
U 1402, 0881,2177,0030,0DD8,0072,2  ;41215         NEXT/CO.NOP                    ; WASTE A CYCLE
                                ;41216
                                ;41217 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41218 =1
                                ;41219 CM.ASR-REG:
                                ;41220         ;1-------------------------------;
U 0071, 0480,05BE,403C,C467,0140,6  ;41221         MDR_R[GPR.R]                   ; GET OPERAND
                                ;41222
                                ;41223 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41224         ;--------------------------------;
U 1406, 0481,2816,403D,8467,0140,A  ;41225         MDR_SEXT(M[MDR]),SIZE[IDEP]    ; SIGN EXTEND TO GET RIGHT SHIFT IN
                                ;41226
                                ;41227         ;--------------------------------;
U 140A, 0580,0EF6,4030,0847,0140,E  ;41228         PL_[1]                         ; CONSTANT FOR LATER
                                ;41229
                                ;41230         ;--------------------------------;
                                ;41231         R[DST.R],SIZ_M[MDR].RR.P,      ; SHIFT RIGHT BY ONE
                                ;41232         CCOP1,SIZE[IDEP],              ; SET CC'S
U 140E, 0483,2177,013C,4847,003F,9  ;41233         IRD1                           ;
```

```
                                ;41234 .TOC     "       Compatibility Mode              : ASL(B)"
                                ;41235
                                ;41236 ;*************************************************************************
                                ;41237 ;       ASL(B)  dest.mx
                                ;41238 ;          Input (dest is memory)       VA      Address of dest
                                ;41239 ;                                       MDR     Contents of dest
                                ;41240 ;
                                ;41241 ;          Input (dest is register)     RNUM    # of register to shift
                                ;41242 ;*************************************************************************
                                ;41243 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41244 =1
                                ;41245 CM.ASL-REG:
                                ;41246         ;1-------------------------------;
U 0077, 0C80,05BE,403C,C467,0141,0 ;41247      MDR_R[GPR.R]                    ; GET SOURCE
                                ;41248
                                ;41249 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41250         ;-------------------------------;
                                ;41251         MTEMP0_ZEXT(M[MDR]),SIZE[IDEP], ; CLEAR POSSIBLE GARBAGE IN REGISTER
U 1410, 0087,259E,4BBC,C047,0141,8 ;41252      RNUM.EQ.7?                      ; SEE IF PC
                                ;41253
                                ;41254 =00
                                ;41255 CM.ASL-SHIFT:
                                ;41256         ;00-------------------------------; NOT REGISTER MODE PC
                                ;41257         R[DST.R].SIZ_M[TEMP0].SL.1,     ; SHIFT LEFT BY ONE
                                ;41258         WRITE NOTREG,                   ; WRITE MEMORY IF NOT REG MODE
                                ;41259         CCOP2,SIZE[IDEP],               ; SET CC'S
U 1418, 0C82,0591,C13C,55DA,003F,9 ;41260      IRD1                            ;
                                ;41261
                                ;41262 =10     ;10-------------------------------; REGISTER MODE PC
                                ;41263         PC_M[PC].SL.1,                  ; SHIFT PC LEFT BY ONE
                                ;41264         CCOP2,SIZE[WORD],               ; SET CC'S
U 141A, 0081,A001,C01D,9487,0148,C ;41265      NEXT/CM.NOP                     ;
                                ;41266
                                ;41267         ;11-------------------------------; NOT REGISTER MODE PC
                                ;41268         R[DST.R].SIZ_M[TEMP0].SL.1,     ; SHIFT LEFT BY ONE
                                ;41269         WRITE NOTREG,                   ; WRITE MEMORY IF NOT REG MODE
                                ;41270         CCOP2,SIZE[IDEP],               ; SET CC'S
U 141B, 0C82,0591,C13C,55DA,003F,9 ;41271      IRD1                            ;
                                ;41272
                                ;41273 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41274 CM.ASL-MEM:
                                ;41275         ;-------------------------------;
U 03D1, 0087,259E,4030,0047,003D,2 ;41276      MTEMP0_ZEXT(M[MDR]),SIZE[IDEP]  ; CLEAR POSSIBLE GARBAGE IN OPERAND
                                ;41277
                                ;41278         ;-------------------------------; NOT REGISTER MODE PC
                                ;41279         R[DST.R].SIZ_M[TEMP0].SL.1,     ; SHIFT LEFT BY ONE
                                ;41280         WRITE NOTREG,                   ; WRITE MEMORY IF NOT REG MODE
                                ;41281         CCOP2,SIZE[IDEP],               ; SET CC'S
U 03D2, 0082,0591,C03C,55DA,0072,2 ;41282      NEXT/CO.NOP                     ;
```

```
                                ;41283 .TOC     ''      Compatibility Mode               : ASH''
                                ;41284
                                ;41285 ;*************************************************************************
                                ;41286 ;       ASH     count.rw,dest.mw
                                ;41287 ;          Input                                 MDR     Count
                                ;41288 ;                                                RNUM    # of register to shift
                                ;41289 ;
                                ;41290 ;       ASH PC will not affect the real pc
                                ;41291 ;       It will instead shift GPR 7 which is undefined in Comp Mode
                                ;41292 ;       This is consistant with the 11/70 and 11/780
                                ;41293 ;*************************************************************************
                                ;41294 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41295 CM.ASH:
                                ;41296         ;-------------------------------;
U 03D3, 0C81,2BC2,403D,8047 0141,3      ;41297         PL_M[MDR]                       ; GET AMOUNT TO SHIFT BY INTO PLATCH
                                ;41298
                                ;41299 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41300         ;-------------------------------;
U 1413, 0886,15BE,403C,C047,0142,9      ;41301         M[TEMP1]_R[GPR.R]               ; GET NUMBER TO SHIFT INTO TEMP1
                                ;41302
                                ;41303 1429:
                                ;41304         ;******FORCE ADDRESS***********;
                                ;41305         M[TEMP1]_SEXT(MB),SIZE[IDEP],   ; SIGN EXTEND OPERAND
U 1429, 0C86,1A56,4DFD,8047,0141,C      ;41306         SHIFT SIZE?,NEXT/CM.ASH.1       ;
                                ;41307
                                ;41308 CM.ASH-CHECKV.-1:
                                ;41309         ;-------------------------------;
U 1416, 0D80,0F76,4030,8047,003E,E      ;41310         SL_[16.]                        ; LOAD SLATCH WITH 16
                                ;41311
                                ;41312 03EE:
                                ;41313         ;*******FORCE ADDRESS***********; GET 16 BITS OF RESULT TO DUAL PORT.
                                ;41314         R[TEMP3].SIZ_Q_Q_D,SIZE[IDEP],  ;
                                ;41315         (PL+SL).GT.32?,                 ;
U 03EE, 0082,002C,7DF0,C047,014B,2      ;41316         NEXT/CM.ASH-CHECKV              ;
                                ;41317
                                ;41318 =00
                                ;41319 CM.ASH.1:
                                ;41320         ;00-------------------------------; PLATCH = 1 THRU 31
                                ;41321         R[GPR.R],SIZ_Q_D_M[TEMP1].ASL.P,; DO  LEFT SHIFT
                                ;41322         CCOP1,SIZE[IDEP],               ; SET N AND Z BITS IN PSL
U 141C, 0C82,1A77,303C,C847,0141,6      ;41323         NEXT/CM.ASH-CHECKV.-1                   ;
                                ;41324
                                ;41325 CM.ASH-STNEG:
                                ;41326         ;01-------------------------------; PLATCH = -1 THRU -31
                                ;41327         R[GPR.R],SIZ_M[TEMP1].ASR.-P,   ; DO THE SHIFT, SET Z AND N
                                ;41328         CCOP1,SIZE[IDEP],               ;
U 141D, 0482,1AB7,003C,C847,0142,2      ;41329         NEXT/CM.ASH-NEG                 ;
                                ;41330
                                ;41331         ;10-------------------------------; PLATCH = 0
                                ;41332         R[GPR.R].SIZ_M[TEMP1],          ; RESTORE ORIGINAL CONTENTS
U 141E, 0082,1592,413C,C847,003F,9      ;41333         CCOP1,SIZE[IDEP],IRD1           ; SO SET CC'S AND LEAVE
                                ;41334
                                ;41335         ;11-------------------------------; PLATCH = -32
                                ;41336         PL_[21],                        ; PL <- -31
U 141F, 0980,0EF6,4031,0847,0141,D      ;41337         NEXT/CM.ASH-STNEG               ; SHIFT BY -31 SAME AS SHIFT BY -32
```

```
                                        ;41338 14B2:
                                        ;41339 CM.ASH-CHECKV:
                                        ;41340          ;0******FORCE ADDRESS**********; PL < 16
                                        ;41341          WB_SEXT(M[TEMP3])-Q,SIZE[IDEP], ; CHECK FOR OVERFLOW
U 14B2, 0480,3818,0A30,0047,0940,8 356* ;41342          WX.EQ.0?,NEXT/CM.ASH-CHECKV.1   ;
                                        ;41343
                                        ;41344 14B3:
                                        ;41345          ;1******FORCE ADDRESS**********; PLATCH >16
U 14B3, 0480,1592,4A30,0047,0940,8 322* ;41346          WB_M[TEMP1],WX.EQ.0?            ; OVERFLOW?
                                        ;41347
                                        ;41348 =0
                                        ;41349 CM.ASH-CHECKV.1:
                                        ;41350          ;0-----------------------------; OVERFLOW
U 1408, 0080,0036,4030,08E7,0140,9      ;41351          SET V                           ; SET V BIT IN PSL
                                        ;41352
                                        ;41353          ;1-----------------------------; NO OVERFLOW
                                        ;41354          WB_D.AND.ZLIT16[1],             ; WB <- D.AND.00010000
U 1409, 0980,0D32,0A70,0847,0140,C      ;41355          WX.NE.0?                        ; CHECK LAST BIT SHIFTED OUT OF GPR
                                        ;41356 =0
                                        ;41357 CM.ASH-CHECKC:
                                        ;41358          ;0-----------------------------; CLEAR C
U 140C, 0080,0036,4130,0047,003F,9      ;41359          IRD1                            ; C-BIT ALREADY CLEAR SO LEAVE
                                        ;41360
                                        ;41361          ;1-----------------------------; SET C
U 140D, 0886,2036,4030,0087,0141,9      ;41362          M[TEMP2]_PSL                    ; READ PSL
                                        ;41363
                                        ;41364 CM.ASH-SETC:
                                        ;41365          ;-------------------------------;
                                        ;41366          PSL_M[TEMP2].OR.ZLIT0[1],       ; SET C-BIT
U 1419, 0180,2C12,4130,0807,003F,9      ;41367          IRDT                            ;
                                        ;41368
                                        ;41369 CM.ASH-NEG:
                                        ;41370
                                        ;41371          ;-------------------------------;
U 1422, 0C81,2BC3,003D,8047,017F,7      ;41372          PL_-M[MDR]                      ; NEGATE PL  ASR.-P DOESN'T WORK RIGHT
                                        ;41373
                                        ;41374 17F7:
                                        ;41375 CM.ASH-TESTC:
                                        ;41376          ;******FORCE ADDRESS**********;
                                        ;41377          WB_M[TEMP1].RR.P,               ; PUT LAST SHIFTED OUT BIT INTO WB-31
U 17F7, 0080,1177,06F0,0047,0140,5      ;41378          WB<31-30>?                      ; TEST CARRIED OUT BIT
                                        ;41379
                                        ;41380 =01      ;01-----------------------------;
U 1405, 0080,0036,4130,0047,003F,9      ;41381          IRD1                            ; C-BIT ALREADY CLEAR SO JUST LEAVE
                                        ;41382
                                        ;41383          ;11-----------------------------;
                                        ;41384          M[TEMP2]_PSL,                   ; GET PSL
U 1407, 0886,2036,4030,0087,0141,9      ;41385          NEXT/CM.ASH-SETC                ; GO SET C-BIT
```

```
                                        ;41386 .TOC     "       Compatibility Mode              : ASHC"
                                        ;41387
                                        ;41388 ;*************************************************************************
                                        ;41389 ;       ASHC    count.rw,dest.mw
                                        ;41390 ;          Input                                 MDR     Count
                                        ;41391 ;                                                RNUM    # of register pair to shift
                                        ;41392 ;
                                        ;41393 ;       ASH PC will not affect the real pc
                                        ;41394 ;       It will instead shift GPR 7 which is undefined in Comp Mode
                                        ;41395 ;       This is consistant with the 11/70 and 11/780
                                        ;41396 ;*************************************************************************
                                        ;41397 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;41398 CM.ASHC:
                                        ;41399         ;-------------------------------;
U 03D4, 0481,2BC2,403D,8047,0142,8      ;41400         PL_M[MDR]                       ; GET SHIFT COUNT
                                        ;41401
                                        ;41402 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                        ;41403         ;-------------------------------;
                                        ;41404         M[TEMP1]_R[GPR.R].RR.SIZ,       ; GET UPPER HALF OF OPERAND
U 1428, 0886,12B7,003C,C047,0142,C      ;41405         SIZE[IDEP]                      ;
                                        ;41406
                                        ;41407         ;-------------------------------;
                                        ;41408         M[TEMP1]_MB.OR.R[GPR.ROR1],     ; CONCAT TWO PARTS OF OPERAND
U 142C, 0086,1A42,4DFE,C047,0142,4      ;41409         SHIFT SIZE?                     ; BRANCH ON COUNT POLARITY AND SIZE
                                        ;41410
                                        ;41411 =00     ;00-----------------------------; PLATCH = 1,31
                                        ;41412         R[TEMP4]_M[TEMP1].ASL.P,        ; DO THE SHIFT
                                        ;41413         SIZE[LONG],CCOP1 SIGND?,        ; SET Z AND N BITS IN PSL
U 1424, 0484,1A77,0B61,0847,0941,5 355* ;41414         NEXT/CM.ASHC-POS                ;
                                        ;41415
                                        ;41416 CM.ASHC-STNEG:
                                        ;41417         ;01-----------------------------; PLATCH = -1,-31
                                        ;41418         R[TEMP4]_M[TEMP1].ASR.-P,       ; DO THE SHIFT
                                        ;41419         SIZE[LONG],CCOP1,SET FLAG1,     ; SET Z AND N BITS IN PSL
U 1425, 0C4C,1AB7,0021,0847,0142,1      ;41420         NEXT/CM.ASHC-NEG                ; SET FLAG TELLING NEG SHIFT
                                        ;41421
                                        ;41422         ;10-----------------------------; PLATCH = 0
                                        ;41423         WB_M[TEMP1],                    ; NOTHING TO DO EXCEPT SET CC'S
                                        ;41424         SIZE[LONG],                     ; SET CC'S ON 32 BITS
U 1426, 0080,1592,4120,0847,003F,9      ;41425         CCOP1,IRD1                      ;
                                        ;41426
                                        ;41427         ;11-----------------------------; PLATCH = -32
                                        ;41428         PL [21],                        ; PLATCH <- -31
U 1427, 0180,0EF6,4031,0847,0142,5      ;41429         NEXT/CM.ASHC-STNEG              ; SHIFT BY -31 IS SAME AS SHIFT BY -32
                                        ;41430 =01
                                        ;41431 CM.ASHC-POS:
                                        ;41432         ;01-----------------------------; RESULT POSITIVE
                                        ;41433         R[TEMP0]_M[TEMP1].ASR.-P,       ; TEMP0 <- SHIFTED OUT BITS'SIGN BIT
                                        ;41434         SET FLAG0,                      ; LEAVE A SIGN TELLING WHERE WE'VE BEEN
U 1415, 0044,1AB7,0A30,0047,0942,0 325* ;41435         WX.EQ.0?,NEXT/CM.ASHC-TESTV     ; IF 0 THEN NO V
                                        ;41436
                                        ;41437         ;11-----------------------------; RESULT NEGATIVE
                                        ;41438         R[TEMP0]_NOT(M[TEMP1].ASR.-P),  ; TEMP0 <- SHIFTED OUT BITS'SIGN BIT
U 1417, 0484,1AB6,0A30,0047,0942,0 325* ;41439         WX.EQ.0?                        ; IF 0 THEN NO V
```

```
                                ;41440 =0
                                ;41441 CM.ASHC-TESTV:
                                ;41442          ;0------------------------------; OVERFLOW
U 1420, 0080,0036,4030,08E7,0142,1      ;41443          SET V                           ; SET V BIT IN PSL
                                ;41444
                                ;41445 CM.ASHC-NEG:
                                ;41446          ;1------------------------------; NO OVERFLOW
                                ;41447          R[GPR.R].SIZ_M[TEMP4].RR.SIZ,   ; WRITE OUT HIGH HALF OF RESULT
U 1421, 0082,43B7,003C,C047,0143,3      ;41448          SIZE[IDEP]                      ;
                                ;41449
                                ;41450          ;-------------------------------;
                                ;41451          R[GPR.ROR1].SIZ_M[TEMP4],       ; WRITE OUT LOW HALF OF RESULT
U 1433, 0482,4592,453E,C047,0143,0      ;41452          SIZE[IDEP],FLAG<1-0>?           ; TEST FOR NEG RESULT
                                ;41453
                                ;41454 =00
                                ;41455 CM.ASHC-TESTC:
                                ;41456          ;00-------------------------------;
U 1430, 0884,0E7F,4030,0047,0143,1      ;41457          R[TEMP0]_RB.XOR.MINUS1          ; COMP THE SHIFTED OUT RESULT
                                ;41458
                                ;41459          ;01-------------------------------;
                                ;41460          WB_R[TEMP0],                    ; FIND LAST BIT SHIFTED OUT
U 1431, 0480,05BE,42B0,0047,0140,C      ;41461          WB<0>?,NEXT/CM.ASH-CHECKC       ; IT'S BIT 0 IN TEMP0
                                ;41462
                                ;41463          ;10-------------------------------;
                                ;41464          PL -M[MDR],                     ; BECAUSE THERE ISN'T A RR.-P
U 1432, 0C81,2BC3,003D,8047,017F,7      ;41465          NEXT/CM.ASH-TESTC               ;
                                ;41466 =
```

```
                                ;41467 .TOC     "       Compatibility Mode               : ADC"
                                ;41468
                                ;*****************************************************************************
                                ;41470 ;       ADC     dest.mx
                                ;41471 ;          Input (dest is memory)        VA      Address of dest
                                ;41472 ;                                        MDR     Contents of dest
                                ;41473 ;
                                ;41474 ;          Input (dest is register)      RNUM    # of register to adc
                                ;41475 ;*****************************************************************************
                                ;41476 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41477 =1
                                ;41478 CM.ADC-REG:
                                ;41479          ;1-------------------------------;
                                ;41480          M[TEMP1]_0,                     ; FREE ZERO TO OR WITH LATER
U 008B, 0C86,15B7,0BBC,C047,0143,8  ;41481      RNUM.EQ.7?                      ; SEE IF SOME IDIOT WANTS TO ADC PC
                                ;41482
                                ;41483 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41484 =00      ;00-----------------------------; NOT REGISTER MODE PC
                                ;41485          R[GPR.R].SIZ_M[TEMP1]+RB+PSLC,  ; ADD CARRY BIT TO REGISTER
U 1438, 0482,10C1,013C,C847,003F,9  ;41486      CCOP1,SIZE[IDEP],IRD1           ; SET CC'S
                                ;41487
                                ;41488 =10      ;10-----------------------------; REGISTER MODE PC
                                ;41489          PC_M[PC]+PSLC,SIZE[WORD],       ; ADD CARRY TO PC
U 143A, 0081,A0C1,029D,8C87,0949,6 325*  ;41490 CCOP1,WB<0>?,NEXT/CM.TEST.PC     ; TEST FOR ODD ADDRESS
                                ;41491
                                ;41492          ;11-----------------------------; NOT REGISTER MODE PC
                                ;41493          R[GPR.R].SIZ_M[TEMP1]+RB+PSLC,  ; ADD CARRY BIT TO REGISTER
U 143B, 0482,10C1,013C,C847,003F,9  ;41494      CCOP1,SIZE[IDEP],IRD1           ; SET CC'S
                                ;41495
                                ;41496 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41497 CM.ADC-MEM:
                                ;41498          ;-------------------------------;
                                ;41499          WRITE M[MDR]+PSLC,SIZE[IDEP],   ; ADD CARRY TO MEMORY LOCATION
U 03D5, 0C61,20C1,003D,85D8,0143,6  ;41500      SET MM.NOINT                    ; DON'T ALLOW RESTART
                                ;41501
                                ;41502 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41503          ;-------------------------------;
                                ;41504          WB_M[MDR]+PSLC,SIZE[IDEP],      ; ADD AGAIN FOR THE SAKE OF SETTING CC'S
U 1436, 0481,20C1,013D,8847,003F,9  ;41505      CCOP1,IRD1                      ;
```

```
                                        ;41506 .TOC     "       Compatibility Mode              : SBC"
                                        ;41507
                                        ;41508 ;*****************************************************************************
                                        ;41509 ;       SBC     dest.mx
                                        ;41510 ;          Input (dest is memory)       VA      Address of dest
                                        ;41511 ;                                       MDR     Contents of dest
                                        ;41512 ;
                                        ;41513 ;          Input (dest is register)     RNUM    # of register to sbc
                                        ;41514 ;*****************************************************************************
                                        ;41515 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;41516 =1
                                        ;41517 CM.SBC-REG:
                                        ;41518         ;1-----------------------------;
                                        ;41519         M[TEMP1]_0,                     ; FREE ZERO TO OR WITH LATER
U 0091, 0C86,15B7,0BBC,C047,0144,0      ;41520         RNUM.EQ.7?                      ; SEE IF SOME IDIOT WANTS TO SBC PC
                                        ;41521
                                        ;41522 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                        ;41523 =00      ;00-----------------------------; NOT REGISTER MODE PC
                                        ;41524         R[GPR.R].SIZ_RB-M[TEMP1]-PSLC,  ; ADD CARRY BIT TO REGISTER
U 1440, 0082,10C3,013C,D047,003F,9      ;41525         CCOP2,SIZE[IDEP],IRD1           ; SET CC'S
                                        ;41526
                                        ;41527 =10      ;10-----------------------------; REGISTER MODE PC
                                        ;41528         PC_M[PC]-PSLC,SIZE[WORD],       ; ADD CARRY TO PC
U 1442, 0481,A0C0,029D,9487,0949,6 325* ;41529         CCOP2,WB<0>?,NEXT/CM.TEST.PC    ; TEST FOR ODD ADDRESS
                                        ;41530
                                        ;41531         ;11-----------------------------; NOT REGISTER MODE PC
                                        ;41532         R[GPR.R].SIZ_RB-M[TEMP1]-PSLC,  ; ADD CARRY BIT TO REGISTER
U 1443, 0082,10C3,013C,D047,003F,9      ;41533         CCOP2,SIZE[IDEP],IRD1           ; SET CC'S
                                        ;41534
                                        ;41535 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;41536 CM.SBC-MEM:
                                        ;41537         ;-------------------------------;
                                        ;41538         WRITE M[MDR]-PSLC,SIZE[IDEP],   ; ADD CARRY TO MEMORY LOCATION
U 03D6, 0861,20C0,003D,85D8,0143,9      ;41539         SET MM.NOINT                    ; DON'T ALLOW RESTART
                                        ;41540
                                        ;41541 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                        ;41542         ;-------------------------------;
                                        ;41543         WB_M[MDR]-PSLC,SIZE[IDEP],      ; ADD AGAIN FOR THE SAKE OF SETTING CC'S
U 1439, 0081,20C0,013D,9047,003F,9      ;41544         CCOP2,IRD1                      ;
```

```
                        ;41545 .TOC     "       Compatibility Mode              : SXT"
                        ;41546
                        ;41547 ;*************************************************************************
                        ;41548 ;       SXT     dest.mx
                        ;41549 ;          Input (dest is memory)       VA      Address of dest
                        ;41550 ;                                       MDR     Contents of dest
                        ;41551 ;
                        ;41552 ;          Input (dest is register)     RNUM    # of register to sxt
                        ;41553 ;
                        ;41554 ;       This instruction has no effect on PC
                        ;41555 ;       STAR does the same thing and the 11/70 always clears PC
                        ;41556 ;*************************************************************************
                        ;41557 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;41558 CM.SXT:
                        ;41559         ;-------------------------------;
                        ;41560         WB PSL,                         ; DRIVE PSL TO BUS TO TEST N-BIT
0 03D7, 0880,0036,4230,0087,0143,7      ;41561         WB<5-0>?                        ; THIS IS THE ONLY WAY TO GET BIT 3
                        ;41562
                        ;41563 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                        ;41564 =110111
                        ;41565         ;110111-------------------------; N-BIT CLEAR
                        ;41566         R[DST.R].SIZ_0,WRITE NOTREG,    ; CLEAR MEMORY OF REGISTER
                        ;41567         CCOP1,SIZE[IDEP],               ; SET CC'S
0 1437, 0482,C5B7,003C,4DDA,0072,2      ;41568         NEXT/CO.NOP                     ; WASTE A CYCLE
                        ;41569
                        ;41570         ;111111-------------------------; N-BIT SET
                        ;41571         R[DST.R].SIZ_-1,WRITE NOTREG,   ; WRITE ALL ONES TO MEMORY OR REGISTER
                        ;41572         CCOP1,SIZE[IDEP],               ; SET CC'S
0 143F, 0082,0E77,003C,4DDA,0072,2      ;41573         NEXT/CO.NOP                     ; WASTE A CYCLE
```

```
                                                ;41574 .TOC     "       Compatibility Mode               : ROL(B)"
                                                ;41575
                                                ;41576 ;*****************************************************************************
                                                ;41577 ;       ROL(B)  dest.mx
                                                ;41578 ;          Input (dest is memory)        VA      Address of dest
                                                ;41579 ;                                        MDR     Contents of dest
                                                ;41580 ;
                                                ;41581 ;          Input (dest is register)      RNUM    # of register to rotate
                                                ;41582 ;*****************************************************************************
                                                ;41583 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                                ;41584 =1
                                                ;41585 CM.ROL-REG:
                                                ;41586         ;1-------------------------------;
U 00A5, 0C80,05BE,403C,C467,0143,E              ;41587         MDR_R[GPR.R]                    ; GET OPERAND
                                                ;41588
                                                ;41589 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                                ;41590         ;-------------------------------;
U 143E, 0081,259D,D030,0047,0144,1              ;41591         Q_ZEXT(M[MDR]).SL.1,SIZE[IDEP]  ; DO INITIAL SHIFT
                                                ;41592
                                                ;41593         ;-------------------------------;
                                                ;41594         MDR_0,                          ; SOMETINNG TO HELP PASS Q LATER
U 1441, 0C80,0036,4B8C,C4E7,0144,4              ;41595         RNUM.EQ.7?                      ; PC ??
                                                ;41596 =00
                                                ;41597 CM.ROL-TEST:
                                                ;41598         ;00-------------------------------; NOT REGISTER MODE PC
                                                ;41599         R[DST.R],SIZ_M[MDR]+Q+PSLC,     ; ADD CARRY TO COMPLETE ROTATE
U 1444, 0083,20C9,013C,5047,003F,9              ;41600         SIZE[IDEP],CCOP2,IRD1           ; SET CC'S
                                                ;41601
                                                ;41602 =10      ;10-------------------------------; REGISTER MODE PC
                                                ;41603         R[TEMP0] ZEXT(M[PC]).SL.1,      ; CLEAR UPPER BITS AND SHIFT
U 1446, 0885,A59D,C010,0047,0144,5              ;41604         SIZE[WORD],NEXT/CM.ROL-PC       ;
                                                ;41605
                                                ;41606         ;11-------------------------------; NOT REGISTER MODE PC
                                                ;41607         R[DST.R],SIZ_M[MDR]+Q+PSLC,     ; ADD CARRY TO COMPLETE ROTATE
U 1447, 0083,20C9,013C,5047,003F,9              ;41608         SIZE[IDEP],CCOP2,IRD1           ; SET CC'S
                                                ;41609
                                                ;41610 CM.ROL-PC:
                                                ;41611         ;-------------------------------;
                                                ;41612         PC_M[TEMP0]+PSLC,               ; ADD CARRY TO COMPLETE ROTATE
                                                ;41613         CCOP2,SIZE[IDEP],               ; SET CC'S
U 1445, 0080,00C1,028D,9487,0949,6 344*         ;41614         WB<0>?,NEXT/CM.TEST.PC          ; TEST PC FOR ODD ADDRESS
                                                ;41615
                                                ;41616 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                                ;41617 CM.ROL-MEM:
                                                ;41618         ;-------------------------------;
U 03D8, C881,259D,D030,0047,0144,9              ;41619         Q_ZEXT(M[MDR]).SL.1,SIZE[IDEP]  ; DO INITIAL SHIFT
                                                ;41620
                                                ;41621 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                                ;41622         ;-------------------------------;
U 1449, 0C80,0036,4030,04E7,0144,D              ;41623         MDR_0                           ; SOMETINNG TO HELP PASS Q LATER
                                                ;41624
                                                ;41625         ;-------------------------------; NOT REGISTER MODE PC
                                                ;41626         R[DST.R],SIZ_M[MDR]+Q+PSLC,     ; ADD CARRY TO COMPLETE ROTATE
                                                ;41627         WRITE NOTREG,SIZE[IDEP],CCOP2,  ; WRITE THE RESULT
U 144D, 0483,20C9,003C,55DA,0072,2              ;41628         NEXT/CO.NOP                     ; WASTE A CYCLE
```

```
                                ;41629 .TOC     "       Compatibility Mode                : ROR(B)"
                                ;41630
                                ;41631 ;*************************************************************************************
                                ;41632 ;       ROR(B)  dest.mx
                                ;41633 ;          Input (dest is memory)        VA      Address of dest
                                ;41634 ;                                        MDR     Contents of dest
                                ;41635 ;
                                ;41636 ;          Input (dest is register)      RNUM    # of register to rotate
                                ;41637 ;*************************************************************************************
                                ;41638 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41639 CM.ROR-MEM:
                                ;41640         ;--------------------------------;
                                ;41641         WB_PSL,                          ; DRIVE PSL TO BUS TO TEST C-BIT
                                ;41642         PL_[1],                          ; CONSTANT FOR LATER
U 03D9, 0580,0EF6,42B0,0887,0143,4  ;41643         WB<0>?                           ; C-BIT IS BIT 0
                                ;41644
                                ;41645 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41646 =0      ;0--------------------------------; C-BIT = 0
                                ;41647         MTEMP0_MDR.AND.OLIT8[255.],      ; TEMP0 <- MDR.AND.FFFEFFF
U 1434, 0987,2F92,0037,F847,0145,4  ;41648         NEXT/CM.ROR-WRITE                ;
                                ;41649
                                ;41650         ;1--------------------------------; C-BIT = 1
U 1435, 0587,2D12,4030,0847,0145,4  ;41651         MTEMP0_MDR.OR.ZLIT16[1]          ; TEMP0 <- MDR.OR.10000
                                ;41652
                                ;41653 CM.ROR-WRITE:
                                ;41654         ;--------------------------------;
                                ;41655         R[DST.R].SIZ_M[TEMP0].RR.P,      ; DO ROTATE AND WRITE
                                ;41656         WRITE NOTREG,SIZE[IDEP],         ; TAKE CARE OF MEMORY MODE
U 1454, 0082,0177,003C,4DDA,0072,2  ;41657         CCOP1,NEXT/CO.NOP                ; SET CC'S - WASTE A CYCLE
                                ;41658
                                ;41659 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41660 CM.RORB-MEM:
                                ;41661         ;--------------------------------;
                                ;41662         WB_PSL,                          ; DRIVE PSL TO BUS TO TEST C-BIT
                                ;41663         PL_[1],                          ; CONSTANT FOR LATER
U 03DA, 0D80,0EF6,42B0,0887,0143,C  ;41664         WB<0>?                           ; C-BIT IS BIT 0
                                ;41665
                                ;41666 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41667 =0      ;0--------------------------------; C-BIT = 0
                                ;41668         MTEMP0_MDR.AND.OLIT8[510.],      ; TEMP0 <- MDR.AND.FFFFFEFF
U 143C, 0987,2F92,003F,F047,0145,4  ;41669         NEXT/CM.ROR-WRITE                ;
                                ;41670
                                ;41671         ;1--------------------------------; C-BIT = 1
                                ;41672         MTEMP0_MDR.OR.ZLIT0[256.],       ; TEMP0 <- MDR.OR.100
U 143D, 0187,2C12,4038,0047,0145,4  ;41673         NEXT/CM.ROR-WRITE                ;
```

```
                                ;41674 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41675 =1
                                ;41676 CM.ROR-REG:
                                ;41677          ;1--------------------------------;
                                ;41678          MDR_ZLIT16[1],                    ; MDR <- 10000
U 00AB, 0580,0D37,0030,0C67,0145,8 ;41679       NEXT/CM.ROR-REG.TESTC             ;
                                ;41680 =1
                                ;41681 CM.RORB-REG:
                                ;41682          ;1--------------------------------;
U 00B1, 0180,0C37,0038,0467,0145,8 ;41683       MDR_ZLIT0[256.]                   ; MDR <- 100
                                ;41684
                                ;41685 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41686 CM.ROR-REG.TESTC:
                                ;41687          ;--------------------------------;
U 1458, 0480,0036,48BC,C047,0144,8 ;41688       RNUM.EQ.7?                        ; TEST FOR PC MODE
                                ;41689
                                ;41690 =00      ;00--------------------------------; NOT REGISTER MODE PC
                                ;41691          WB_PSL,                           ; DRIVE PSL TO BUS TO TEST C-BIT
                                ;41692          PL_[1],                           ; CONSTANT FOR LATER
U 1448, 0D80,0EF6,42B0,0887,0147,4 ;41693       WB<0>?,NEXT/CM.ROR-REG.RESULT     ; TEST C-BIT
                                ;41694
                                ;41695 =10      ;10--------------------------------; REGISTER MODE PC
                                ;41696          WB_PSL,                           ; DRIVE PSL TO BUS TO TEST C-BIT
                                ;41697          PL_[1],                           ; CONSTANT FOR LATER
U 144A, 0D80,0EF6,42B0,0887,0147,E ;41698       WB<0>?,NEXT/CM.ROR-PC.RESULT      ; TEST C-BIT
                                ;41699
                                ;41700          ;11--------------------------------; NOT REGISTER MODE PC
                                ;41701          WB_PSL,                           ; DRIVE PSL TO BUS TO TEST C-BIT
                                ;41702          PL_[1],                           ; CONSTANT FOR LATER
U 144B, 0D80,0EF6,42B0,0887,0147,4 ;41703       WB<0>?                            ; TEST C-BIT
                                ;41704
                                ;41705 =0
                                ;41706 CM.ROR-REG.RESULT:
                                ;41707          ;0--------------------------------; C-BIT CLEAR
                                ;41708          MTEMP0_MDR.NOTAND.R[GPR.R],       ; SET OR CLEAR BIT TO BE ROTATED TO MSB
U 1474, 0887,2003,C03C,C047,0145,4 ;41709       NEXT/CM.ROR-WRITE                 ;
                                ;41710
                                ;41711          ;1--------------------------------; C-BIT SET
                                ;41712          MTEMP0_MDR.OR.R[GPR.R],           ; SET OR CLEAR BIT TO BE ROTATED TO MSB
U 1475, 0887,2002,403C,C047,0145,4 ;41713       NEXT/CM.ROR-WRITE                 ;
                                ;41714
                                ;41715 =0
                                ;41716 CM.ROR-PC.RESULT:
                                ;41717          ;0--------------------------------; C-BIT CLEAR
                                ;41718          MTEMP0_PC.ANDNOT.R[TEMP1],        ; SET OR CLEAR BIT TO BE ROTATED TO MSB
U 147E, 0487,A003,8030,4047,0145,A ;41719       NEXT/CM.ROR-PC.WRITE              ;
                                ;41720
                                ;41721          ;1--------------------------------; C-BIT SET
U 147F, 0087,A002,4030,4047,0145,A ;41722       MTEMP0_PC.OR.R[TEMP1]             ; SET OR CLEAR BIT TO BE ROTATED TO MSB
                                ;41723
                                ;41724 CM.ROR-PC.WRITE:
                                ;41725          ;--------------------------------;
                                ;41726          PC_M[TEMP0],RR.P,                 ; WRITE RESULT TO PC
                                ;41727          CCOP1,SIZE[IDEP],                 ; SET CC'S
U 145A, 0880,0177,02B0,0C87,0149,6 ;41728       WB<0>?,NEXT/CM.TEST.PC            ; TEST FOR ODD ADDRESS
```

```
                                    ;41729 .TOC     "       Compatibility Mode                : SWAB"
                                    ;41730
                                    ;41731 ;*****************************************************************************
                                    ;41732 ;       SWAB    dest.mx
                                    ;41733 ;          Input (dest is memory)        VA      Address of dest
                                    ;41734 ;                                        MDR     Contents of dest
                                    ;41735 ;
                                    ;41736 ;          Input (dest is register)      RNUM    # of register to swab
                                    ;41737 ;*****************************************************************************
                                    ;41738 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                    ;41739 =1
                                    ;41740 CM.SWAB-REG:
                                    ;41741          ;1-------------------------------;
U 008B, 0480,058E,403C,C467,0145,C  ;41742          MDR_R[GPR.R]                    ; GET FIRST OPERAND
                                    ;41743
                                    ;41744 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                    ;41745          ;-------------------------------;
U 145C, 0885,2637,0030,0047,0145,E  ;41746          R[TEMP0]_M[MDR].BCDSWP          ; SWAP BYTES AND PLACE IN SUPROT 31-16
                                    ;41747
                                    ;41748          ;-------------------------------;
                                    ;41749          RNUM.EQ.7?,                     ; TEST FOR PC
U 145E, 0C80,0036,4BBC,C047,0144,C  ;41750          NEXT/CM.SWAB-RUN                ;
                                    ;41751
                                    ;41752 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                    ;41753 CM.SWAB-MEM:
                                    ;41754          ;-------------------------------;
U 03DB, 0885,2637,0030,0047,0144,C  ;41755          R[TEMP0]_M[MDR].BCDSWP          ; SWAP BYTES AND PLACE IN SUPROT 31-16
                                    ;41756
                                    ;41757 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                    ;41758 =00
                                    ;41759 CM.SWAB-RUN:
                                    ;41760          ;00------------------------------; NOT REGISTER MODE PC
                                    ;41761          R[DST.R].SIZ_Q_M[TEMP0].RR.SIZ, ; ROTATE TO RIGHT POSITION ON WAY TO REG
                                    ;41762          WRITE NOTREG,SIZE[WORD],        ; WRITE OUT DEST
U 144C, 0082,03B7,101C,45DA,0146,2  ;41763          NEXT/CM.SWAB-SET.CC             ;
                                    ;41764
                                    ;41765 =10      ;10------------------------------; REGISTER MODE PC
                                    ;41766          R[TEMP0]_M[PC].BCDSWP,          ; SWAP BYTES AND PLACE IN SUPROT 31-16
U 144E, 0485,A637,0030,0047,0146,5  ;41767          NEXT/CM.SWAB-PC                 ;
                                    ;41768
                                    ;41769          ;11------------------------------; NOT REGISTER MODE PC
                                    ;41770          R[DST.R].SIZ_Q_M[TEMP0].RR.SIZ, ; ROTATE TO RIGHT POSITION ON WAY TO REG
                                    ;41771          WRITE NOTREG,SIZE[WORD],        ; WRITE OUT DEST
U 144F, 0082,03B7,101C,45DA,0146,2  ;41772          NEXT/CM.SWAB-SET.CC             ;
```

```
                                        ;41773 CM.SWAB-SET.CC:
                                        ;41774          ;--------------------------------;
                                        ;41775          WB_Q                             ; DRIVE DEST TO WB
                                        ;41776          SIZE[BYTE],CCOP1,                ; SETE CC'S ON LOWER BYTE OF DEST
U 1462, 0480,003A,410D,8847,003F,9      ;41777          IRD1                             ;
                                        ;41778
                                        ;41779 CM.SWAB-PC:
                                        ;41780          ;--------------------------------;
                                        ;41781          PC_Q_M[TEMP0].RR.SIZ,            ; ROTATE TO RIGHT POSITION ON WAY TO PC
U 1465, 0880,03B7,1010,0487,0146,9      ;41782          SIZE[WORD]                       ; SET CC'S
                                        ;41783
                                        ;41784          ;--------------------------------;
                                        ;41785          WB_Q                             ; DRIVE DEST TO WB
                                        ;41786          SIZE[BYTE],CCOP1,                ; SET CC'S ON LOWER BYTE OF DEST
U 1469, 0080,003A,428D,8847,0149,6      ;41787          WB<0>?,NEXT/CM.TEST.PC           ; TEST PC FOR ODD ADDRESS
```

```
                                ;41788 .TOC     "       Compatibility Mode                : MOV(B)"
                                ;41789
                                ;41790 ;*************************************************************************************
                                ;41791 ;       MOV(B)  src.rx,dest.wx
                                ;41792 ;          Input (dest is memory)        VA      Address of dest
                                ;41793 ;                                        MDR     Source
                                ;41794 ;
                                ;41795 ;          Input (dest is register)      RNUM    # of register to move to
                                ;41796 ;*************************************************************************************
                                ;41797 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41798 =0
                                ;41799 CM.MOVB-REG:
                                ;41800          ;0-------------------------------;
                                ;41801          Q_SEXT(M[MDR]),SIZE[BYTE],      ; DO A SIGN EXTEND FOR MOVB SRC,REG
U 033A, 0081,2816,500D,8047,0033,C ;41802       NEXT/CM.MOV-WRITE               ;
                                ;41803
                                ;41804          ;1-------------------------------;
                                ;41805          PC_SEXT(M[MDR]).                ; MOVE TO PC
                                ;41806          CCOP1,SIZE[BYTE],               ; SET CC'S
U 033B, 0881,2816,428D,8C87,0149,6 ;41807       WB<0>?,NEXT/CM.TEST.PC          ; TEST FOR ODD ADDRESS
                                ;41808
                                ;41809 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41810 =1
                                ;41811 CM.MOV-REG:
                                ;41812          ;1-------------------------------;
                                ;41813          Q_M[MDR],                       ; DO THIS TO MAKE COMMON CODE POSSIBLE
U 00C5, 0481,2592,5BBC,C047,0033,C ;41814       RNUM.EQ.7?                      ; MOV   SRC,PC ???
                                ;41815
                                ;41816 =00
                                ;41817 CM.MOV-WRITE:
                                ;41818          ;00------------------------------; NOT REGISTER MODE PC
                                ;41819          R[DST.R],SIZ_Q_Q_D,             ; WRITE TO REG
                                ;41820          WRITE NOTREG,CCOP1,             ; SET CC'S
U 033C, 0C82,002C,711C,4DDA,003F,9 ;41821       SIZE[WORD],IRD1                 ; ALWAYS WRITE WHOLE REGISTER
                                ;41822
                                ;41823 =10      ;10------------------------------; REGISTER MODE PC
                                ;41824          PC_Q,                           ; MOVE TO PC
                                ;41825          CCOP1,SIZE[WORD],               ; SET CC'S
U 033E, 0C80,003A,429D,8C87,0149,6 ;41826       WB<0>?,NEXT/CM.TEST.PC          ; TEST FOR ODD ADDRESS
                                ;41827
                                ;41828          ;11------------------------------; NOT REGISTER MODE PC
                                ;41929          R[DST.R],SIZ_Q_Q_D,             ; WRITE TO REG
                                ;41830          WRITE NOTREG,CCOP1,             ; SET CC'S
U 033F, 0C82,002C,711C,4DDA,003F,9 ;41831       SIZE[WORD],IRD1                 ; ALWAYS WRITE WHOLE REGISTER
                                ;41832
                                ;41833 CM.MOV-MEM:
                                ;41834          ;-------------------------------;
                                ;41835          R[DST.R]_Q_W_D,                 ; WRITE TO MEMORY
                                ;41836          WRITE NOTREG,SIZE[IDEP],CCOP1,  ; SET CC'S
U 03DC, 0484,002C,703C,4DDA,0072,2 ;41837       NEXT/CO.NOP                     ; WASTE A CYCLE
```

```
                                        ;41838 .TOC     ""      Compatibility Mode              : ADD""
                                        ;41839
                                        ;41840 ;*************************************************************************************
                                        ;41841 ;       ADD     src.rw,dest.mw
                                        ;41842 ;          Input (dest is memory)       VA      Address of dest
                                        ;41843 ;                                       QREG    Source
                                        ;41844 ;                                       MDR     Contents of dest
                                        ;41845 ;
                                        ;41846 ;          Input (dest is register)     MDR     Source
                                        ;41847 ;                                       RNUM    # of register to add to
                                        ;41848 ;*************************************************************************************
                                        ;41849 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;41850 CM.ADD-MEM:
                                        ;41851         ;-------------------------------;
                                        ;41852         WRITE M[MDR]+Q,                 ; DO THE ADDITION AND WRITE TO MEMORY
                                        ;41853         CCOP1,SIZE[IDEP],               ; SET CC'S
U 03DD, 0C81,2009,0030,0DD8,0072,2      ;41854         NEXT/CO.NOP                     ; WASTE A CYCLE
                                        ;41855 =0
                                        ;41856 CM.ADD-REG:
                                        ;41857         ;0------------------------------; NOT REGISTER MODE PC
                                        ;41858         R[DST.R].SIZ_M[MDR]+RB,         ; DO ADD AND WRITE REGISTER
                                        ;41859         WRITE NOTREG,SIZE[IDEP],CCOP1,  ; SET CC'S
U 0342, 0C83,2001,013C,4DDA,003F,9      ;41860         IRD1                            ;
                                        ;41861
                                        ;41862         ;1------------------------------; REGISTER MODE PC
U 0343, 0881,A592,5030,0047,0146,D      ;41863         Q_M[PC]                         ; GET PC TO ADD TO MDR
                                        ;41864
                                        ;41865 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                        ;41866         ;-------------------------------;
                                        ;41867         PC_M[MDR]+Q,                    ; DO 16 BIT ADD TO PC
                                        ;41868         CCOP1,SIZE[IDEP],               ; SET CC'S
U 146D, 0481,2009,0280,0C87,0949,6 337* ;41869         WB<0>?.NEXT/CM.TEST.PC          ; TEST PC FOR ODD ADDRESS
```

```
                                        ;41870 .TOC   ''       Compatibility Mode                : SUB''
                                        ;41871
                                        ;41872 ;*****************************************************************************
                                        ;41873 ;       SUB     src.rw,dest.mw
                                        ;41874 ;          Input (dest is memory)        ;A      Address of dest
                                        ;41875 ;                                        QREG    Source
                                        ;41876 ;                                        MDR     Contents of dest
                                        ;41877 ;
                                        ;41878 ;          Input (dest is register)      MDR     Source
                                        ;41879 ;                                        RNUM    # of register to sub from
                                        ;41880 ;*****************************************************************************
                                        ;41881 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;41882 CM.SUB-MEM:
                                        ;41883         ;--------------------------------;
                                        ;41884         WRITE M[MDR]-Q,                  ; DO SUB AND PUT BACK TO MEMORY
                                        ;41885         CCOP2,SIZE[IDEP],                ; SET CC'S
U 03DE, 0881,2008,0030,15D8,0072,2      ;41886         NEXT/CO.NOP                      ; WASTE A CYCLE
                                        ;41887 =0
                                        ;41888 CM.SUB-REG:
                                        ;41889         ;0-------------------------------; NOT REGISTER MODE PC
                                        ;41890         R[DST.R].SIZ_RB-M[MDR],          ; DO SUB AND WRITE TO REG
                                        ;41891         WRITE NOTREG,SIZE[IDEP],CCOP2,   ; SET CC'S
U 034A, 0883,2003,013C,55DA,003F,9      ;41892         IRD1                             ;
                                        ;41893
                                        ;41894         ;1-------------------------------; REGISTER MODE PC
U 034B, 0081,A592,5030,0047,0147,1      ;41895         Q_M[PC]                          ; GET PC TO SUB FROM MDR
                                        ;41896
                                        ;41897 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                        ;41898         ;--------------------------------;
                                        ;41899         PC_Q-M[MDR],                     ; DO SUB FROM PC
                                        ;41900         CCOP2,SIZE[IDEP],                ; SET CC'S
U 1471, 0081,200B,0280,1487,0949,6 337* ;41901         WB<0>?,NEXT/CM.TEST.PC           ; TEST PC FOR ODD ADDRESS
```

```
                                        ;41902 .TOC     ''      Compatibility Mode              : CMP(B)''
                                        ;41903
                                        ;41904 ;*************************************************************************************
                                        ;41905 ;       CMP(B)  src1.rx,src2.mx
                                        ;41906 ;          Input (src2 is memory)       QREG    Source1
                                        ;41907 ;                                       MDR     Source2
                                        ;41908 ;
                                        ;41909 ;          Input (src2 is register)     MDR     Source1
                                        ;41910 ;                                       RNUM    # of register containing Source2
                                        ;41911 ;*************************************************************************************
                                        ;41912 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;41913 CM.CMP-MEM:
                                        ;41914         ;-------------------------------;
                                        ;41915         WB_Q-M[MDR],                    ; DO THE COMPARE TO SET CC'S
                                        ;41916         CCOP2,SIZE[IDEP],               ; SET CC'S
U 03DF, 0881,200B,0030,1047,0072,2      ;41917         NEXT/CO.NOP                     ; WASTE A CYCLE
                                        ;41918 =0
                                        ;41919 CM.CMP-REG:
                                        ;41920         ;0------------------------------; NOT REGISTER MODE PC
                                        ;41921         WB_M[MDR]-R[GPR.R],             ; DO THE COMPARE
                                        ;41922         CCOP2,SIZE[IDEP],               ; SET CC'S
U 034C, 0081,2000,013C,D047,003F,9      ;41923         IRD1                            ;
                                        ;41924
                                        ;41925         ;1------------------------------; REGISTER MODE PC
U 034D, 0885,A592,4035,C047,0147,9      ;41926         R[R7]_M[PC]                     ; GET PC
                                        ;41927
                                        ;41928 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                        ;41929         ;-------------------------------;
                                        ;41930         WB_M[MDR]-R[R7],                ; DO THE COMPARE WITH SAVED PC
                                        ;41931         CCOP2,SIZE[IDEP],               ; SET CC'S
U 1479, 0081,2000,0135,D047,003F,9      ;41932         IRD1                            ;
```

```
                                ;41933 .TOC     "       Compatibility Mode              : MUL"
                                ;41934
                                ;41935 ;*************************************************************************************
                                ;41936 ;       MUL  prod.mw,mulr.rw
                                ;41937 ;          Input                                MDR     Multiplicand
                                ;41938 ;                                               RNUM    # of reg to contain product
                                ;41939 ;
                                ;41940 ;       MUL PC will not affect the real pc
                                ;41941 ;       It will instead use GPR 7 which is undefined in Comp Mode
                                ;41942 ;       This is consistant with the 11/70 and 11/780
                                ;41943 ;*************************************************************************************
                                ;41944 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;41945 CM.MUL:
                                ;41946         ;-------------------------------;
                                ;41947         R[TEMP6] SEXT(M[MDR]),          ; SEXT MULTIPLICAND
U 03E0, 0C85,2E5E,0031,98C7,0140,0 ;41948      SIZE[IDEP],ALUS_SIGND           ; SAVE IT'S SIGN
                                ;41949
                                ;41950 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;41951 =000     ;000----------------------------;
                                ;41952         M[TEMP4] R[GPR.R]               ; MULTIPLIER
                                ;41953         CMP SIGNS?,SIZE[I EP],          ; BUT TO THE PROPER PLACE IN MUL SUB
U 1400, 0C86,45BE,4B7C,C8C7,0C03,0 340* ;41954 PUSH,NEXT/IL.MULSUB.MDR_0         ;
                                ;41955
                                ;41956 =011     ;011----------------------------; RETURN +3        POS PRODUCT
                                ;41957         WB_D+ALKC,                      ; RESULT LESS THAN 2**15?
                                ;41958         SIGND CMP?,SIZE[IDEP],          ; EQUAL TO 0 ???
U 1403, 0C80,0061,0B7D,8047,0942,E 380* ;41959 NEXT/CM.MUL-POS                  ;
                                ;41960
                                ;41961 =100     ;100----------------------------; RETURN +4        NEG PRODUCT
                                ;41962         WB_D+ALKC,                      ; RESULT MORE THAN -2**15?
                                ;41963         SIGND CMP?,SIZE[IDEP],          ; EQUAL TO 0 ???
U 1404, 0480,0061,0B7D,8047,0942,A 380* ;41964 NEXT/CM.MUL-NEG                  ;
                                ;41965 =
                                ;41966 =10
                                ;41967 CM.MUL-NEG:
                                ;41968         ;10-----------------------------; NO,RESULT EQUAL OR LESS THAN -2**15
                                ;41969         WB_D,WX.EQ.0?,                  ;
U 142A, 0C80,0022,4A3D,8047,017F,2 ;41970      NEXT/CM.MUL-NEG.WRT2            ;
                                ;41971
                                ;41972         ;11-----------------------------; YES,RESULT MORE THAN -2**15
                                ;41973         M[TEMP1] R[ZERO]-Q,WX.EQ.0?,    ; GET 2'S COMP OF LOW HALF OF RESULT
U 142B, 0886,1038,0A3D,8047,097F,0 407* ;41974 NEXT/CM.MUL-NEG.WRT3             ;
                                ;41975
                                ;41976 17F2:
                                ;41977 CM.MUL-NEG.WRT2:
                                ;41978         ;0-----------------------------; LESS THAN -2**15
                                ;41979         M[TEMP1] R[ZERO]-Q,             ;
                                ;41980         SET FLAGT,                      ;
U 17F2, 084E,1038,003D,8047,017F,0 ;41981      NEXT/CM.MUL-NEG.WRT3            ;
                                ;41982 17F3:
                                ;41983         ;1-----------------------------; LESS OR EQUAL TO -2**15
                                ;41984         WB_(R[ZERO]+Q).SL.1,SIZE[IDEP], ;
                                ;41985         SIGND CMP?,                     ;
U 17F3, 0880,0039,CB7D,8047,083F,A 391* ;41986 NEXT/CM.MUL-NEG.WRT              ;
```

B	1	MICRO2	1M(01)	J	5	MICRO2	1M(01)	E	10	MICRO2	1M(01)
C	1	MICRO2	1M(01)	K	5	MICRO2	1M(01)	F	10	MICRO2	1M(01)
D	1	MICRO2	1M(01)	L	5	MICRO2	1M(01)	G	10	MICRO2	1M(01)
E	1	MICRO2	1M(01)	M	5	MICRO2	1M(01)	H	10	MICRO2	1M(01)
F	1	MICRO2	1M(01)	N	5	MICRO2	1M(01)	I	10	MICRO2	1M(01)
G	1	MICRO2	1M(01)	B	6	MICRO2	1M(01)	J	10	MICRO2	1M(01)
H	1	MICRO2	1M(01)	C	6	MICRO2	1M(01)	K	10	MICRO2	1M(01)
I	1	MICRO2	1M(01)	D	6	MICRO2	1M(01)	L	10	MICRO2	1M(01)
J	1	MICRO2	1M(01)	E	6	MICRO2	1M(01)	M	10	MICRO2	1M(01)
K	1	MICRO2	1M(01)	F	6	MICRO2	1M(01)	N	10	MICRO2	1M(01)
L	1	MICRO2	1M(01)	G	6	MICRO2	1M(01)	B	11	MICRO2	1M(01)
M	1	MICRO2	1M(01)	H	6	MICRO2	1M(01)	C	11	MICRO2	1M(01)
N	1	MICRO2	1M(01)	I	6	MICRO2	1M(01)	D	11	MICRO2	1M(01)
B	2	MICRO2	1M(01)	J	6	MICRO2	1M(01)	E	11	MICRO2	1M(01)
C	2	MICRO2	1M(01)	K	6	MICRO2	1M(01)	F	11	MICRO2	1M(01)
D	2	MICRO2	1M(01)	L	6	MICRO2	1M(01)	G	11	MICRO2	1M(01)
E	2	MICRO2	1M(01)	M	6	MICRO2	1M(01)	H	11	MICRO2	1M(01)
F	2	MICRO2	1M(01)	N	6	MICRO2	1M(01)	I	11	MICRO2	1M(01)
G	2	MICRO2	1M(01)	B	7	MICRO2	1M(01)	J	11	MICRO2	1M(01)
H	2	MICRO2	1M(01)	C	7	MICRO2	1M(01)	K	11	MICRO2	1M(01)
I	2	MICRO2	1M(01)	D	7	MICRO2	1M(01)	L	11	MICRO2	1M(01)
J	2	MICRO2	1M(01)	E	7	MICRO2	1M(01)	M	11	MICRO2	1M(01)
K	2	MICRO2	1M(01)	F	7	MICRO2	1M(01)	N	11	MICRO2	1M(01)
L	2	MICRO2	1M(01)	G	7	MICRO2	1M(01)	B	12	MICRO2	1M(01)
M	2	MICRO2	1M(01)	H	7	MICRO2	1M(01)	C	12	MICRO2	1M(01)
N	2	MICRO2	1M(01)	I	7	MICRO2	1M(01)	D	12	MICRO2	1M(01)
B	3	MICRO2	1M(01)	J	7	MICRO2	1M(01)	E	12	MICRO2	1M(01)
C	3	MICRO2	1M(01)	K	7	MICRO2	1M(01)	F	12	MICRO2	1M(01)
D	3	MICRO2	1M(01)	L	7	MICRO2	1M(01)	G	12	MICRO2	1M(01)
E	3	MICRO2	1M(01)	M	7	MICRO2	1M(01)	H	12	MICRO2	1M(01)
F	3	MICRO2	1M(01)	N	7	MICRO2	1M(01)	I	12	MICRO2	1M(01)
G	3	MICRO2	1M(01)	B	8	MICRO2	1M(01)	J	12	MICRO2	1M(01)
H	3	MICRO2	1M(01)	C	8	MICRO2	1M(01)	K	12	MICRO2	1M(01)
I	3	MICRO2	1M(01)	D	8	MICRO2	1M(01)	L	12	MICRO2	1M(01)
J	3	MICRO2	1M(01)	E	8	MICRO2	1M(01)	M	12	MICRO2	1M(01)
K	3	MICRO2	1M(01)	F	8	MICRO2	1M(01)	N	12	MICRO2	1M(01)
L	3	MICRO2	1M(01)	G	8	MICRO2	1M(01)	B	13	MICRO2	1M(01)
M	3	MICRO2	1M(01)	H	8	MICRO2	1M(01)	C	13	MICRO2	1M(01)
N	3	MICRO2	1M(01)	I	8	MICRO2	1M(01)	D	13	MICRO2	1M(01)
B	4	MICRO2	1M(01)	J	8	MICRO2	1M(01)	E	13	MICRO2	1M(01)
C	4	MICRO2	1M(01)	K	8	MICRO2	1M(01)	F	13	MICRO2	1M(01)
D	4	MICRO2	1M(01)	L	8	MICRO2	1M(01)	G	13	MICRO2	1M(01)
E	4	MICRO2	1M(01)	M	8	MICRO2	1M(01)	H	13	MICRO2	1M(01)
F	4	MICRO2	1M(01)	N	8	MICRO2	1M(01)	I	13	MICRO2	1M(01)
G	4	MICRO2	1M(01)	B	9	MICRO2	1M(01)	J	13	MICRO2	1M(01)
H	4	MICRO2	1M(01)	C	9	MICRO2	1M(01)	K	13	MICRO2	1M(01)
I	4	MICRO2	1M(01)	D	9	MICRO2	1M(01)	L	13	MICRO2	1M(01)
J	4	MICRO2	1M(01)	E	9	MICRO2	1M(01)	M	13	MICRO2	1M(01)
K	4	MICRO2	1M(01)	F	9	MICRO2	1M(01)	N	13	MICRO2	1M(01)
L	4	MICRO2	1M(01)	G	9	MICRO2	1M(01)	B	14	MICRO2	1M(01)
M	4	MICRO2	1M(01)	H	9	MICRO2	1M(01)	C	14	MICRO2	1M(01)
N	4	MICRO2	1M(01)	I	9	MICRO2	1M(01)	D	14	MICRO2	1M(01)
B	5	MICRO2	1M(01)	J	9	MICRO2	1M(01)	E	14	MICRO2	1M(01)
C	5	MICRO2	1M(01)	K	9	MICRO2	1M(01)	F	14	MICRO2	1M(01)
D	5	MICRO2	1M(01)	L	9	MICRO2	1M(01)	G	14	MICRO2	1M(01)
E	5	MICRO2	1M(01)	M	9	MICRO2	1M(01)	H	14	MICRO2	1M(01)
F	5	MICRO2	1M(01)	N	9	MICRO2	1M(01)	I	14	MICRO2	1M(01)
G	5	MICRO2	1M(01)	B	10	MICRO2	1M(01)	J	14	MICRO2	1M(01)
H	5	MICRO2	1M(01)	C	10	MICRO2	1M(01)	K	14	MICRO2	1M(01)
I	5	MICRO2	1M(01)	D	10	MICRO2	1M(01)	L	14	MICRO2	1M(01)

M	14	MICRO2	1M(01)
N	14	MICRO2	1M(01)
B	15	MICRO2	1M(01)
C	15	MICRO2	1M(01)
D	15	MICRO2	1M(01)
E	15	MICRO2	1M(01)
F	15	MICRO2	1M(01)
G	15	MICRO2	1M(01)
H	15	MICRO2	1M(01)
I	15	MICRO2	1M(01)
J	15	MICRO2	1M(01)
K	15	MICRO2	1M(01)
L	15	MICRO2	1M(01)
M	15	MICRO2	1M(01)
N	15	MICRO2	1M(01)
B	16	MICRO2	1M(01)
C	16	MICRO2	1M(01)
D	16	MICRO2	1M(01)
E	16	MICRO2	1M(01)
F	16	MICRO2	1M(01)
G	16	MICRO2	1M(01)
H	16	MICRO2	1M(01)
I	16	MICRO2	1M(01)
J	16	MICRO2	1M(01)
K	16	MICRO2	1M(01)
L	16	MICRO2	1M(01)

```
                                ;41987 03FA:
                                ;41988 CM.MUL-NEG.WRT:
                                ;41989          ;10-----------------------------; RESULT LESS THAN -2**15
                                ;41990          M[TEMP0] R[ZERO]-D-ALKC,        ; GET 2'S COMP OF HIGH HALF RESULT
U 03FA, 004E,0063,003D,8047,017F,2 ;41991       SET FLAGT,NEXT/CM.MUL-NEG.WRT2
                                ;41992
                                ;41993 03FB:
                                ;41994          ;11-----------------------------; Q MOST NEG NO OVERFLOW
                                ;41995          M[TEMP1] R[ZERO]-Q,             ; GET 2'S COMP OF HIGH HALF RESULT
U 03FB, 0086,1038,003D,8047,017F,0 ;41996       NEXT/CM.MUL-NEG.WRT3            ;
                                ;41997
                                ;41998
                                ;41999 17F0:
                                ;42000 CM.MUL-NEG.WRT3:
                                ;42001          ;0------------------------------; RESULT NOT EQUAL TO ZERO
                                ;42002          M[TEMP0] R[ZERO]-D-ALKC,        ; GET TWO'S COMPLIMENT OF HIGH HALF
U 17F0, 0086,0063,003D,8047,0142,D ;42003       NEXT/CM.MUL-NEG.WRT1
                                ;42004
                                ;42005 17F1:
                                ;42006          ;1------------------------------; RESULT EQUAL TO ZERO
                                ;42007          M[TEMP0] R[ZERO]-D-ALKC,        ; GET TWO'S COMPLIMENT OF HIGH HALF
                                ;42008          CLEAR FLAG0,                    ; RESET FLAG0
U 17F1, 0806,0063,003D,8047,0142,D ;42009       NEXT/CM.MUL-NEG.WRT1
                                ;42010
                                ;42011 142D:
                                ;42012 CM.MUL-NEG.WRT1:
                                ;42013          ;******FORCE ADDRESS************;
                                ;42014          R[GPR.R].SIZ_M[TEMP0],          ; WRITE HIGH HALF OF RESULT
U 142D, 0082,0592,403C,C847,014B,1 ;42015       CCOP1,SIZE[IDEP]                ; SET Z IF ZERO
                                ;42016
                                ;42017 14B1:
                                ;42018          ;******FORCE ADDRESS************;
                                ;42019          R[GPR.ROR1].SIZ_M[TEMP1],       ; WRITE LOW HALF OF RESULT
                                ;42020          CCOP2,SIZE[IDEP],               ; RESET Z BIT IF NECESSARY
U 14B1, 0882,1592,403E,D047,0148,5 ;42021       NEXT/CM.MUL-READ.PSW            ;
                                ;42022
                                ;42023 =10
                                ;42024 CM.MUL-POS:
                                ;42025          ;10-----------------------------; NO, LARGER OR EQUAL 2**15
                                ;42026          R[GPR.R].SIZ_D,                 ; WRITE HIGH HALF OF RESULT
                                ;42027          SET FLAG1,                      ; SAVE OVERFLOW INFO FOR LATER
                                ;42028          CCOP1,SIZE[IDEP],               ; SET Z BIT IF = 0
U 142E, 044A,05B2,403C,C847,0142,3 ;42029       NEXT/CM.MUL-POS.WRT2            ;
                                ;42030
                                ;42031          ;11-----------------------------; YES, LESS THAN 2**15
                                ;42032          R[GPR.R].SIZ_D,                 ; WRITE HIGH HALF OF RESULT
U 142F, 0C82,05B2,403C,C847,0142,3 ;42033       CCOP1,SIZE[IDEP]                ; SET Z BIT IF = 0
                                ;42034 1423:
                                ;42035 CM.MUL-POS.WRT2:
                                ;42036          ;******FORCE ADDRESS************;
                                ;42037          R[GPR.ROR1].SIZ_M[MDR].OR.Q,    ; WRITE LOW HALF OF RESULT
U 1423, 0883,200A,403E,D047,0148,5 ;42038       CCOP2,SIZE[IDEP]                ; RESET Z BIT
```

```
                                ;42039 CM.MUL-READ.PSW:
                                ;42040          ;-------------------------------;
U 1485, 0086,2036,4030,0087,0148,B ;42041       M[TEMP2]_PSL                    ; GET PARTIALLY SET CC'S
                                ;42042
                                ;42043          ;-------------------------------;
                                ;42044          M[TEMP2]_MB.AND.OLITO[502.],    ; CLEAR N AND C BITS OF PSW
U 148B, 0186,2E12,053F,B047,0145,0 ;42045       FLAG<1-0>?                      ; BRANCH ON FLAGS TO SET CC'S
                                ;42046
                                ;42047 =00      ;00-----------------------------;
                                ;42048          PSL_M[TEMP2],                   ; CLEAR N AND C
U 1450, 0880,2002,413D,8007,003F,9 ;42049       IRDT                            ;
                                ;42050
                                ;42051          ;01-----------------------------;
                                ;42052          PSL_M[TEMP2].OR.ZLITO[8],       ; SET N CLEAR C
U 1451, 0180,2C12,4130,4007,003F,9 ;42053       IRDT                            ;
                                ;42054
                                ;42055          ;10-----------------------------;
                                ;42056          PSL_M[TEMP2].OR.ZLITO[1],       ; CLEAR N SET C
U 1452, 0180,2C12,4130,0807,003F,9 ;42057       IRDT                            ;
                                ;42058
                                ;42059          ;11-----------------------------;
                                ;42060          PSL_M[TEMP2].OR.ZLITO[9],       ; SET N SET C
U 1453, 0580,2C12,4130,4807,003F,9 ;42061       IRDT                            ;
```

```
                                ;42062 .TCC     ''      Compatibility Mode              : DIV''
                                ;42063
                                ;42064 ;*************************************************************************
                                ;42065 ;       DIV  dend.mw,dsor.rw
                                ;42066 ;          Input                                MDR     Divisor
                                ;42067 ;                                               RNUM    # of reg to contain Dividend
                                ;42068 ;
                                ;42069 ;       DIV PC will not affect the real pc
                                ;42070 ;       It will instead use GPR 7 which is undefined in Comp Mode
                                ;42071 ;       This is consistant with the 11/70 and 11/780
                                ;42072 ;*************************************************************************
                                ;42073 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42074 CM.DIV:
                                ;42075          ;-------------------------------;
                                ;42076          M[TEMP1]_R[GPR.R],              ; GET REGISTER TO MBUS TO ZEXT
U 03E1, 08AE,15BE,403C,C047,014B,0 ;42077       STEPC_6                         ; TEST FOR OVERFLOW SET IN LAST CYCLE
                                ;42078
                                ;42079 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;42080          ;-------------------------------;
                                ;42081          R[TEMP3]_SEXT(M[MDR]),          ; GET SEXT DIVISOR TO TEMP3 FOR DIVSUB
                                ;42082          ALUS_SIGND,SIZE[IDEP],          ; GET SIGN FOR ENTRY TO DIVSUB
U 14B0, 0085,2E5E,0A70,D8C7,0148,0 ;42083       WX.NE.0?                        ; CHECK FOR DIVIDE BY ZERO
                                ;42084
                                ;42085 =0       ;0-------------------------------; DIVIDE BY ZERO
                                ;42086          CC_[7],                         ; SET Z,V AND C BITS IN PSW
U 1480, 0980,0C37,0130,38A7,003F,9 ;42087       IRD1                            ; EXIT QUIETLY
                                ;42088
                                ;42089          ;1-------------------------------;
                                ;42090          Q_R[GPR.ROR1] M[TEMP0]_0,       ; GET LOW HALF OF DIVIDEND
U 1481, 0C86,058C,703E,C047,0145,7 ;42091       NEXT/CM.DIV-CALL.SUB            ; PROVIVE ZERO FOR DIVSUB
                                ;42092
                                ;42093 =00
                                ;42094 =01
                                ;42095 CM.DIV-FIX.REM:
                                ;42096          ;01------------------------------; RETURN -2 QUO WILL BE +
                                ;42097          WB_Q-R[ZERO],                   ; IF RESULT NOT POSITIVE THEN OVERFLOW
                                ;42098          SIGND CMP?,SIZE[IDEP],          ;
U 1455, 0480,003B,087D,8047,0945,D 436* ;42099  NEXT/CM.DIV-TEST.OV1             ;
                                ;42100
                                ;42101          ;10------------------------------; RETURN -1 QUO WILL BE -
                                ;42102          R[TEMP0]_Q_-Q,SIZE[IDEP],       ; NEGATE Q
U 1456, 0C84,0038,187D,8047,0948,8 436* ;42103  SIGND CMP?,NEXT/CM.DIV-TEST.OV   ; OVERFLOW IF POSITIVE (NOT EQUAL 0)
                                ;42104
                                ;42105 CM.DIV-CALL.SUB:
                                ;42106          ;11------------------------------;
                                ;42107          M[TEMP1]_D_SEXT(MB),            ; ZEXT HIGH HALF OF DIVIDEND
                                ;42108          CMP SIGNS?,SIZE[IDEP],PUSH,     ; SAVE SIGN OF DIVIDEND IN MTEMP1<15>
U 1457, 0056,1816,687D,88C7,0444,8 ;42109       SET FLAG2,NEXT/IL.DIVSUB        ; SET EDIV FLAG, BUT INTO DIVSUB
```

```
                                ;42110 14B8:
                                ;42111 CM.DIV-TEST.OV:
                                ;42112          ;00*****FORCE ADDRESS**********; OVERFLOW OR MOST NEGATIVE
                                ;42113          WB_M[TEMP0].AND.ZLIT8[80],      ; TEST FOR MOST NEGATIVE QUOTIENT
U 14B8, 0580,0D92,0A34,0047,014B,A ;42114       WX.EQ.0?,NEXT/CM.DIV-TEST.OV2   ;
                                ;42115
                                ;42116 14B9:
                                ;42117          ;01*****FORCE ADDRESS**********; NO OVERFLOW (QUOTIENT ZERO)
                                ;42118          M[TEMP5]_D_REM,                 ; UNSHIFT REMAINDER
U 14B9, 0C86,5026,AB7D,9847,0146,1 ;42119       ALUS?,NEXT7CM.DIV-TEST.REM1     ; TEST FOR UNCORRECTED REMAINDER
                                ;42120
                                ;42121 14BA:
                                ;42122 CM.DIV-TEST.OV2:
                                ;42123          ;10*****FORCE ADDRESS**********; NO OVERFLOW
                                ;42124          M[TEMP5]_D_REM,                 ; UNSHIFT REMAINDER
U 14BA, 0C86,5026,AB7D,9847,0146,1 ;42125       ALUS?,NEXT7CM.DIV-TEST.REM1     ; TEST FOR UNCORRECTED REMAINDER
                                ;42126
                                ;42127 14BB:
                                ;42128          ;11*****FORCE ADDRESS**********; OVERFLOW!
U 14BB, 0080,0036,4030,08E7,0148,2 ;42129       SET V,NEXT/CM.DIV-CLEAR.C       ; SET V BIT, CLEAR C BIT
                                ;42130 =01
                                ;42131 CM.DIV-TEST.OV1:
                                ;42132          ;01----------------------------; NO OVERFLOW
                                ;42133          M[TEMP5]_D_REM,                 ; UNSHIFT REMAINDER
U 145D, 0C86,5026,AB7D,9847,0146,1 ;42134       ALUS?,NEXT7CM.DIV-TEST.REM1     ; TEST FOR UNCORRECTED REMAINDER
                                ;42135
                                ;42136          ;11----------------------------; OVERFLOW !!
                                ;42137          SET V,                          ; SET V BIT
U 145F, 0050,0036,4030,08E7,0148,2 ;42138       NEXT/CM.DIV-CLEAR.C             ; GO CLEAR C-BIT
                                ;42139
                                ;42140 =00
                                ;42141 CM.DIV-TEST.REM:
                                ;42142          ;00----------------------------; DIVISOR NEG, REDO CORRECTION
U 1460, 0C80,5000,2030,C047,0146,1 ;42143       D_M[TEMP5]-R[TEMP3]             ; D <- UNCORRECTED REM - DIVISOR
                                ;42144
                                ;42145 CM.DIV-TEST.REM1:
                                ;42146          ;01----------------------------;
                                ;42147          R[GPR.R].SIZ_Q_Q_D,             ; WRITE OUT DIVIDEND
                                ;42148          CCOP1,SIZE[IDEP],               ; SET CC'S
                                ;42149          DIVIDEND SIGN?,                 ; TEST DIVIDEND SIGN IN MTEMP1<15>
U 1461, 0082,102C,7ABC,C847,0145,9 ;42150       NEXT/CM.DIV-END                 ;
                                ;42151
                                ;42152 =11      ;11----------------------------; CORRECT REM (ASSUME DIVISOR POS)
                                ;42153          D_D+R[TEMP3],                   ; D <- UNCORRECTED REM + DIVISOR
U 1463, 0080,0021,24F0,C047,0146,0 ;42154       FLAG3?,NEXT/CM.DIV-TEST.REM     ; TEST FOR DIVISOR POS
```

```
                                        ;42155 1459:
                                        ;42156 CM.DIV-END:
                                        ;42157          ;01*****FORCE ADDRESS**********; DIVIDEND NEGATIVE
                                        ;42158          R[GPR.ROR1].SIZ_-D,SIZE[IDEP],  ; WRITE OUT REMAINDER AS NEG VALUE
U 1459, 0482,05B3,003E,C047,0148,2      ;42159          NEXT/CM.DIV-CLEAR.C             ; GO CLEAR C-BIT
                                        ;42160
                                        ;42161 145B:
                                        ;42162          ;11*****FORCE ADDRESS**********; DIVIDEND POSITIVE
                                        ;42163          R[GPR.ROR1].SIZ_D,SIZE[IDEP],   ; WRITE OUT REMAINDER AS POS VALUE
U 145B, 0482,05B2,403E,C047,0148,2      ;42164          NEXT/CM.DIV-CLEAR.C             ; GO CLEAR C-BIT
                                        ;42165
                                        ;42166
                                        ;42167
                                        ;42168 1482:
                                        ;42169 CM.DIV-CLEAR.C:
                                        ;42170          ;*******FORCE ADDRESS**********; (ADDRESS 1488 MOVED IN REV 18)
U 1482, 0C86,0036,4030,0087,0148,3      ;42171          M[TEMP0]_PSL                    ; GET PSL
                                        ;42172
                                        ;42173 1483:
                                        ;42174          ;*******FORCE ADDRESS**********; (ADDRESS 1489 MOVED IN REV 18)
                                        ;42175          PSL_M[TEMP0].ANDNOT.ZLIT0[1],   ; CLEAR C-BIT IN PSL
U 1483, 0180,0C13,8130,0807,003F,9      ;42176          IRDT                            ;
```

```
                                ;42177 .TOC     ""       Compatibility Mode                : XOR"
                                ;42178
                                ;42179 ;*****************************************************************************
                                ;42180 ;       XOR     src.rw,dest.mw
                                ;42181 ;          Input                                RNUM    Register to XOR
                                ;42182 ;
                                ;42183 ;*****************************************************************************
                                ;42184 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42185 =1111
                                ;42186 CM.XOR:
                                ;42187          ;1111-----------------------------;
U 013F, 0480,0036,4BBC,C047,0034,4  ;42188      RNUM.EQ.7?                          ; TEST FOR PC AS SOURCE
                                ;42189
                                ;42190 =00      ;00-----------------------------;
                                ;42191          MDR_R[GPR.R],                       ; SAVE CONTENTS OF SOURCE REGISTER
U 0344, 0880,05BE,407C,C467,0000,1  ;42192      IRDX [1]                            ; JUMP INTO OS.MOD
                                ;42193
                                ;42194 =10      ;10-----------------------------;
                                ;42195          MDR_M[PC],                          ; SAVE PC AS SOURCE REGISTER
U 0346, 0881,A002,407D,8467,0000,1  ;42196      IRDX [1]                            ; JUMP INTO OS.MOD
                                ;42197
                                ;42198          ;11-----------------------------;
                                ;42199          MDR_R[GPR.R],                       ; SAVE CONTENTS OF SOURCE REGISTER
U 0347, 0880,05BE,407C,C467,0000,1  ;42200      IRDX [1]                            ; JUMP INTO OS.MOD
                                ;42201
                                ;42202 CM.XOR-WRITE:
                                ;42203          ;-----------------------------;
U 03E2, 0C80,0036,44B0,0047,0077,8  ;42204      FLAG2?                              ; TEST FOR REGISTER MODE
                                ;42205
                                ;42206 0778:    ;*********FORCE ADDRESS*********;
                                ;42207 CM.XOR-WRITE1:
                                ;42208          ;000----------------------------;
                                ;42209          R[DST.R].SIZ_M[MDR].XOR.Q,          ; DO XOR AND WRITE RESULT
                                ;42210          WRITE NOTREG,SIZE[IDEP],CCOP1,      ; WRITE MEM IF NECESSARY
U 0778, 0483,200B,403C,4DDA,0072,2  ;42211      NEXT/CO.NOP                         ; SET CC'S
                                ;42212
                                ;42213 077A:    ;*********FORCE ADDRESS*********;
                                ;42214          ;010----------------------------;
                                ;42215          PC_M[MDR].XOR.Q,                    ; WRITE PC
                                ;42216          SIZE[IDEP],CCOP1,                   ; SET CC'S
U 077A, 0C81,200B,4280,0C87,0149,6  ;42217      WB<0>?,NEXT/CM.TEST.PC              ; CHECK FOR ODD ADDRESS
                                ;42218
                                ;42219 077B:    ;011----------------------------;
                                ;42220          R[DST.R].SIZ_M[MDR].XOR.Q,          ; DO XOR AND WRITE RESULT
                                ;42221          WRITE NOTREG,SIZE[IDEP],CCOP1,      ; WRITE MEM IF NECESSARY
U 077B, 0483,200B,403C,4DDA,0072,2  ;42222      NEXT/CO.NOP                         ; SET CC'S
                                ;42223
                                ;42224 077C:    ;100----------------------------;
                                ;42225          RNUM.EQ.7?,                         ; TEST FOR PC AS DEST
U 077C, 0C80,0036,4BBC,C047,0077,8  ;42226      NEXT/CM.XOR-WRITE1                  ;
```

```
                                        ;42227 .TOC     "       Compatibiiity Mode              : BIS(B)"
                                        ;42228
                                        ;42229 ;*****************************************************************************
                                        ;42230 ;       BIS(B)  src.rx,dst.mx
                                        ;42231 ;          Input (dest is memory)       VA      Address of dest
                                        ;42232 ;                                       QREG    Source
                                        ;42233 ;                                       MDR     Contents of dest
                                        ;42234 ;
                                        ;42235 ;          Input (dest is register)     MDR     Source
                                        ;42236 ;                                       RNUM    # of register to bis
                                        ;42237 ;*****************************************************************************
                                        ;42238 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;42239 CM.BIS-MEM:
                                        ;42240         ;-------------------------------;
                                        ;42241         WRITE M[MDR].OR.Q,              ; SET SPECIFIED BITS AND WRITE
                                        ;42242         CCOP1,SIZE[IDEP],               ; SET CC'S
U 03E3, 0881,200A,4030,0DD8,0072,2      ;42243         NEXT/CO.NOP                     ; WASTE A CYCLE
                                        ;42244 =0
                                        ;42245 CM.BIS-REG:
                                        ;42246         ;0------------------------------; NOT REGISTER MODE PC
                                        ;42247         R[GPR.R].SIZ_M[MDR].OR.RB,      ; SET BITS AND PUT IN REGISTER
                                        ;42248         CCOP1,SIZE[IDEP],               ; SET CC'S
U 034E, 0483,2002,413C,C847,003F,9      ;42249         IRD1                            ;
                                        ;42250
                                        ;42251         ;1------------------------------; REGISTER MODE PC
U 034F, 0085,A592,4030,0047,014B,7      ;42252         R[TEMP0]_M[PC]                  ; GET DEST OPERAND
                                        ;42253
                                        ;42254 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                        ;42255 14B7:
                                        ;42256         ;******FORCE ADDRESS*********; (14B7 FROM 14BA TO CM.DIV IN REV 18)
                                        ;42257         PC_M[MDR].OR.R[TEMP0],          ; WRITE ANSWER TO PC
                                        ;42258         SIZE[IDEP],CCOP1,               ; SET CC'S
U 14B7, 0C81,2002,42B0,0C87,0149,6      ;42259         WB<0>?,NEXT/CM.TEST.PC          ;
```

```
                        ;42260 .TOC   "       Compatibility Mode                : BIT(B)"
                        ;42261
                        ;42262 ;*************************************************************************
                        ;42263 ;       BIT(B)  src.rx,dst.mx
                        ;42264 ;          Input (dest is memory)        VA      Address of dest
                        ;42265 ;                                        QREG    Source
                        ;42266 ;                                        MDR     Contents of dest
                        ;42267 ;
                        ;42268 ;          Input (dest is register)      MDR     Source
                        ;42269 ;                                        RNUM    # of register to bit
                        ;42270 ;*************************************************************************
                        ;42271 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                        ;42272 CM.BIT-MEM:
                        ;42273         ;--------------------------------;
                        ;42274         WB_M[MDR].AND.Q,                ; AND TWO NUMBERS
                        ;42275         CCOP1,SIZE[IDEP],               ; SET CC'S
U 03E4, 0081,2002,0130,0847,003F,9      ;42276         IRD1                            ;
                        ;42277 =0
                        ;42278 CM.BIT-REG:
                        ;42279         ;0-------------------------------; NOT REGISTER MODE PC
                        ;42280         WB_M[MDR].AND.R[GPR.R],         ; DO TEST
                        ;42281         CCOP1,SIZE[IDEP],               ; SET CC'S
U 0352, 0481,2002,013C,C847,003F,9      ;42282         IRD1                            ;
                        ;42283
                        ;42284         ;1-------------------------------; REGISTER MODE PC
U 0353, 0885,A592,4030,0047,003E,A      ;42285         R[TEMP0]_M[PC]                  ; GET DEST OPERAND
                        ;42286
                        ;42287 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                        ;42288 03EA:
                        ;42289         ;********FORCE ADDRESS***********;
                        ;42290         WB_M[MDR].AND.R[TEMP0],         ; TEST ANSWER
                        ;42291         SIZE[IDEP],CCOP1,               ; SET CC'S
U 03EA, 0481,2002,0130,0847,003F,9      ;42292         IRD1                            ;
```

```
                                ;42293 .TOC     "       Compatibility Mode              : BIC(B)"
                                ;42294
                                ;42295 ;*****************************************************************************
                                ;42296 ;       BIC(B)  src.rx,dst.mx
                                ;42297 ;          Input (dest is memory)       VA      Address of dest
                                ;42298 ;                                       QREG    Source
                                ;42299 ;                                       MDR     Contents of dest
                                ;42300 ;
                                ;42301 ;          Input (dest is register)     MDR     Source
                                ;42302 ;                                       RNUM    # of register to bic
                                ;42303 ;*****************************************************************************
                                ;42304 .REGION/IRDX.R1L,!RDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42305 CM.BIC-MEM:
                                ;42306         ;-------------------------------;
                                ;42307         WRITE M[MDR].ANDNOT.Q,          ; CLEAR SPECIFIED BITS AND WRITE
                                ;42308         CCOP1,SIZE[IDEP],               ; SET CC'S
J 03E5, 0C81,200B,8030,0DD8,0072,2 ;42309      NEXT/CO.NOP                     ; WASTE A CYCLE
                                ;42310 =0
                                ;42311 CM.BIC-REG:
                                ;42312         ;0-------------------------------; NOT REGISTER MODE PC
                                ;42313         R[GPR.R].SIZ_M[MDR].NOTAND.RB,  ; CLEAR BITS AND PUT IN REGISTER
                                ;42314         CCOP1,SIZE[IDEP],               ; SET CC'S
U 0354, 0483,2003,C13C,C847,003F,9 ;42315      IRD1                            ;
                                ;42316
                                ;42317         ;1-------------------------------; REGISTER MODE PC
U 0355, 0885,A592,4030,0047,014B,6 ;42318      R[TEMP0]_M[PC]                  ; GET DEST OPERAND
                                ;42319
                                ;42320 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;42321         ;-------------------------------;
                                ;42322         PC_M[MDR].NOTAND.R[TEMP0],                      ; WRITE ANSWER TO PC
                                ;42323         SIZE[IDEP],CCOP1,               ; SET CC'S
U 1486, 0C81,2003,C2B0,0C87,0149,6 ;42324      WB<0>?,NEXT/CM.TEST.PC          ;
```

```
                                ;42325 .TOC     "       Compatibility Mode               : BR, BXXX"
                                ;42326
                                ;42327 ;*************************************************************************
                                ;42328 ;       BR      disp.bb
                                ;42329 ;          Input                                 OSR     Disp
                                ;42330 ;*************************************************************************
                                ;42331 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42332 =1110
                                ;42333 CM.BR:
                                ;42334         ;1110-------------------------;
                                ;42335         MDR_ZEXT(OSR),                ; GET BRANCH DISPLACEMENT
U 014F, 0480,0036,4030,05E7,0148,D ;42336      NEXT/CM.BRCND-BRANCH          ; GO DO BRANCH
                                ;42337 =
                                ;42338
                                ;42339 ;*************************************************************************
                                ;42340 ;       BXXX  disp.bb
                                ;42341 ;          Input                                 OSR     Disp
                                ;42342 ;*************************************************************************
                                ;42343 =1110
                                ;42344 CM.BRCND:
                                ;42345         ;1110-------------------------;
U 015E, 0C80,0036,4B70,05E7,0148,C ;42346      MDR_ZEXT(OSR) BRATST?         ; GET BRANCH DISPLACEMENT
                                ;42347 =                                      ; TEST FOR BRANCH
                                ;42348
                                ;42349 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;42350 =0
                                ;42351 CM.NOP: ;0-------------------------------; BRANCH CONDITION FALSE
U 148C, 0080,0036,4130,0047,093F,9 ;42352      IRD1                            ; NOTHING TO DO   FALL THROUGH BRANCH
                                ;42353
                                ;42354 CM.BRCND-BRANCH:
                                ;42355         ;1--------------------------------; BRANCH CONDITION TRUE
                                ;42356         PC_PC+(SEXT(M[MDR]).SL.1),      ; PC <- PC+DISP*2
U 148D, 0881,2815,C03D,8587,0148,C ;42357      SIZE[IDEP],NEXT/CM.NOP          ; WASTE A CYCLE TO LET PC CATCH UP
```

```
                                ;42358 .TOC     "       Compatibility Mode               : JSR"
                                ;42359
                                ;42360 ;*****************************************************************************
                                ;42361 ;       JSR     reg,dest.aw
                                ;42362 ;          Input                                 VA      Address of subroutine
                                ;42363 ;                                                RNUM    # of register to save
                                ;42364 ;*****************************************************************************
                                ;42365 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42366 CM.JSR:
                                ;42367          ;-------------------------------;
                                ;42368          FLAGS?,                         ; TEST FOR REGISTER MODE
U 03E6, 0080,0036,44F0,0047,0148,E      ;42369          NEXT/CM.JSR-TEST.MODE           ; RESERVED OPERAND FAULT IF SO
                                ;42370
                                ;42371 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;42372 =00
                                ;42373 CM.JSR-MODE.OK:
                                ;42374          ;00-----------------------------; NOT REGISTER MODE PC
                                ;42375          WRITE R[GPR.R],SIZE[IDEP],      ; SAVE REGISTER ON STACK
U 1464, 0C80,05BE,403C,C5D8,014C,2      ;42376          NEXT/CM.JSR-WRITE.REG           ;
                                ;42377
                                ;42378 =10      ;10-----------------------------; REGISTER MODE PC
                                ;42379          R[TEMP1]_M[PC],                 ; SAVE PC FOR LATER
U 1466, 0485,A592,4030,4047,014C,3      ;42380          NEXT/CM.JSR-WRITE.PC            ;
                                ;42381
                                ;42382          ;11-----------------------------; NOT REGISTER MODE PC
                                ;42383          WRITE R[GPR.R],SIZE[IDEP],      ; SAVE REGISTER ON STACK
U 1467, 0C80,05BE,403C,C5D8,014C,2      ;42384          NEXT/CM.JSR-WRITE.REG           ;
                                ;42385
                                ;42386 CM.JSR-WRITE.REG:
                                ;42387          ;-------------------------------;
                                ;42388          R[GPR.R].SIZ_M[PC],SIZE[IDEP],  ; SAVE PC IN REG FOR RET
U 14C2, 0883,A592,403C,C047,0001,0      ;42389          NEXT/CM.JSR-WRITE.PC.1          ; SKIP ONE
                                ;42390
                                ;42391 CM.JSR-WRITE.PC:
                                ;42392          ;-------------------------------;
U 14C3, 0C80,1592,4030,05D8,0001,0      ;42393          WRITE M[TEMP1],SIZE[IDEP]       ; PUSH PC
                                ;42394 0010:
                                ;42395 CM.JSR-WRITE.PC.1:
                                ;42396          ;*********FORCE ADDRESS*********;
                                ;42397          PC_M[TEMP0],                    ; STUFF PC WITH SUBROUTINE ADDRESS
U 0010, 0880,0002,42BD,8487,0149,6      ;42398          WB<0>?,NEXT/CM.TEST.PC          ; GO DO IRD1
                                ;42399 1670:
                                ;42400 CM.JSR-TEST.SP:
                                ;42401          ;0********FORCE ADDRESS*********;
                                ;42402          VA_R6_RB-CONX(2) RNUM.EQ.7?,    ; DEC SP BEFORE PUSHING REGISTER
                                ;42403          PUSH RBS-,                      ; SAVE STATE INCASE OF BAD THINGS
U 1670, 0885,473C,0B95,84A7,0146,4      ;42404          NEXT/CM.JSR-MODE.OK             ;
                                ;42405 1671:
                                ;42406          ;1********FORCE ADDRESS*********;
U 1671, 0C80,0036,4030,0047,00FD,9      ;42407          NEXT/IE.CM.ODD                  ; CAUSE ODD ADDRESS TRAP
                                ;42408 =0
                                ;42409 CM.JSR-TEST.MODE:
                                ;42410          ;0-------------------------------;
                                ;42411          WB_R[R6],WB<0>?,                ; TEST FOR ODD
U 148E, 0080,05BE,42B5,8047,0167,0      ;42412          NEXT/CM.JSR-TEST.SP             ;
```

```
                                        ;42413
                                        ;42414          ;1--------------------------------;
U 148F, 0480,0036,4031,0047,00FB,E      ;42415          NEXT/IE.CM.ILLINS                  ; ERROR TIME
```

```
                                                    ;42416 .TOC     "       Compatibility Mode                : RTS, JMP"
                                                    ;42417
                                                    ;42418 ;*****************************************************************************
                                                    ;42419 ;       RTS  reg
                                                    ;42420 ;          Inputs                              RNUM    # of register to restore
                                                    ;42421 ;*****************************************************************************
                                                    ;42422 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                                    ;42423 =1111
                                                    ;42424 CM.RTS:
                                                    ;42425         ;1111-----------------------------;
                                                    ;42426         M[TEMP1]_R[R6],MDR_ZEXT(OSR),     ; SET UP TO POP WORD FROM STACK
                                                    ;42427         SET FLAGT,WB<0>?,                 ; CHECK FOR ODD SP
U 014F, 044E,15BE,42B5,85E7,014D,0                  ;42428         NEXT/CM-TEST.OSR                  ; AND GO CHECK FOR RESERVED OPCODES
                                                    ;42429
                                                    ;42430 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                                    ;42431 =00
                                                    ;42432 CM.RTS.CONTINUE:
                                                    ;42433         ;00-----------------------------; NOT REGISTER MODE PC
                                                    ;42434         PC_R[GPR.R],                      ; SET PC TO RETURN ADDRESS
U 1468, 0480,05BE,42BC,C487,014C,4                  ;42435         WB<0>?,NEXT/CM.RTS-REST.REG       ;
                                                    ;42436
                                                    ;42437 =10      ;10-----------------------------; REGISTER MODE PC
                                                    ;42438         PC_ZEXT(M[MDR]),SIZE[IDEP],       ; RESTORE PC FROM STACK
U 146A, 0881,2016,42BD,8487,0149,6                  ;42439         WB<0>?,NEXT/CM.TEST.PC            ; WASTE A CYCLE TO LET PC SETTLE
                                                    ;42440
                                                    ;42441         ;11-----------------------------; NOT REGISTER MODE PC
                                                    ;42442         PC_R[GPR.R],                      ; SET PC TO RETURN ADDRESS
U 146B, 0480,05BE,42BC,C487,014C,4                  ;42443         WB<0>?,NEXT/CM.RTS-REST.REG       ;
                                                    ;42444 14C4:
                                                    ;42445 CM.RTS-REST.REG:
                                                    ;42446         ;0********FORCE ADDRESS*********;
                                                    ;42447         R[GPR.R].SIZ_M[MDR],SIZE[IDEP],   ; RESTORE REGISTER FROM STACK
U 14C4, 0083,2592,413C,C047,003F,9                  ;42448         IRD1                              ;
                                                    ;42449
                                                    ;42450 14C5:    ;1********FORCE ADDRESS*********;
U 14C5, 0C80,0036,4030,0047,0149,7                  ;42451         NEXT/CM.TEST.PC.ODD               ;
```

```
                                ;42452 ;*****************************************************************************
                                ;42453 ;       JMP     dest.aw
                                ;42454 ;          Inputs                              VA      Address to jump to
                                ;42455 ;*****************************************************************************
                                ;42456 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42457 =1
                                ;42458 CM.JMP-REG:
                                ;42459          ;1-------------------------------;
U 00D1, 0480,0036,4030,0047,00FB,E  ;42460      NEXT/IE.CM.ILLINS                 ; CAN'T JUMP TO A REGISTER
                                ;42461
                                ;42462 CM.JMP-MEM:
                                ;42463          ;-------------------------------;
                                ;42464          PC_M[VA],                         ; DO THE JUMP
U 03E7, 0881,B002,42BD,8487,0149,6  ;42465      WB<0>?,NEXT/CM.TEST.PC            ; WASTE SOME TIME
```

```
                                ;42466 .TOC     ''      Compatibility Mode              : SOB, CLx, SEx''
                                ;42467
                                ;42468 ;*****************************************************************************
                                ;42469 ;       SOB     reg,dest.ab
                                ;42470 ;          Inputs                                OSR     Branch displacement
                                ;42471 ;                                                RNUM    # of register to sob
                                ;42472 ;*****************************************************************************
                                ;42473 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42474 =1111
                                ;42475 CM.SOB:
                                ;42476         ;1111-------------------------;
                                ;42477         R[GPR.R].SIZ_RB-1,              ; DEC THE COUNT REGISTER
                                ;42478         MDR_ZEXT(OSR),SIZE[IDEP],       ; GET DISPLACEMENT INCASE OF BRANCH
U 015F, 0C82,0E7D,0A7C,C5E7,0949,0 372*  ;42479         WX.NE.0?                        ; TEST FOR ZERO
                                ;42480
                                ;42481 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;42482 =0       ;0-------------------------;
U 1490, 0080,0036,4130,0047,003F,9       ;42483         IRD1                            ; SOB ALL DONE...   FALL THROUGH
                                ;42484
                                ;42485         ;1-------------------------;
U 1491, 0181,2C12,1031,F847,014C,6       ;42486         Q_M[MDR].AND.ZLIT0[3F]          ; MASK OFF BAD BITS IN DISP
                                ;42487
                                ;42488         ;-------------------------;
U 14C6, 0C86,0038,003D,8047,014C,7       ;42489         M[TEMP0]_R[ZERO]-Q              ; MTEMP0 <- -DISP
                                ;42490
                                ;42491         ;-------------------------;
                                ;42492         PC_PC+(M[TEMP0].SL.1),          ; PC <- PC-DISP*2
U 14C7, 0880,0E52,C030,0587,0148,C       ;42493         NEXT/CM.NOP                     ;
                                ;42494
                                ;42495 ;*****************************************************************************
                                ;42496 ;       CLx, SEx
                                ;42497 ;          Inputs                                OSR     Info on which bits to set/clear
                                ;42498 ;*****************************************************************************
                                ;42499 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42500 =1111
                                ;42501 CM-SET.CLEAR:
                                ;42502         ;-------------------------;
U 016F, 0C80,0036,4030,05E7,014C,8       ;42503         MDR_ZEXT(OSR)                   ; GET PATTERN TO USE INTO PSW
                                ;42504
                                ;42505 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;42506         ;-------------------------;
                                ;42507         M[TEMP0]_PSL,                   ; GET THE CURRENT PSL
U 14C8, 0C86,0036,48B0,0087,0149,2       ;42508         IR<2>?                          ; BRANCH ON WHETHER SET OR CLEAR BITS
                                ;42509
                                ;42510 =0       ;0-------------------------;
                                ;42511         CC_M[MDR].NOTAND.R[TEMP0],      ; CLEAR THE SPECIFIED CC BITS
U 1492, 0C81,2003,C130,00A7,003F,9       ;42512         IRD1                            ;
                                ;42513
                                ;42514         ;1-------------------------;
                                ;42515         CC_M[MDR].OR.R[TEMP0],          ; SET THE SPECIFIED CC BITS
U 1493, 0C81,2002,4130,00A7,003F,9       ;42516         IRD1                            ;
```

```
                                        .TOC    ''      Compatibility Mode              : MFPI, MFPD''

                                        ;*************************************************************************
                                        ;       MFPx    src.rw
                                        ;          Input (src is memory)        MDR     src
                                        ;
                                        ;          Input (src is register)      RNUM    # of register to PUSH
                                        ;*************************************************************************
                                        .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        =1
                                        CM.MFPI-REG:
                                        CM.MFPD-REG:
                                                ;1-----------------------------;
                                                WB_R[R6],WB<0>?,               ; TEST FOR ODD SP
U 00D7, 0080,05BE,42B5,8047,0166,8              NEXT/CM.MFPD-TEST.SP           ;

                                        .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                        =00
                                        CM.MFPD-TEST.MODE:
                                                ;00----------------------------; NOT REGISTER MODE PC
                                                WRITE R[GPR.R],SIZE[WORD],     ; PUSH REGISTER
U 146C, 0C80,05BE,401C,CDD8,0072,2              CCOP1,NEXT/CO.NOP              ; SET CC'S

                                        =10     ;10----------------------------; REGISTER MODE PC
                                                R[TEMPO]_M[PC],                ; GET PC
U 146E, 0085,A592,4010,0847,014C,9              CCOP1,SIZE[WORD],NEXT/CM.MFP-PC ; SET CC'S

                                                ;11----------------------------; NOT REGISTER MODE PC
                                                WRITE R[GPR.R],SIZE[WORD],     ; PUSH REGISTER
U 146F, 0C80,05BE,401C,CDD8,0072,2              CCOP1,NEXT/CO.NOP              ; SET CC'S

                                        CM.MFP-PC:
                                                ;------------------------------;
                                                WRITE M[TEMPO],SIZE[WORD],     ; PUSH PC
U 14C9, 0C80,0592,4010,05D8,0072,2              NEXT/CO.NOP                    ;

                                        .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        CM.MFPD-MEM:
                                        CM.MFPI-MEM:
                                                ;------------------------------;
                                                VA_R[R6]_RB-CONX(2),           ; DEC STACK BEFORE PUSHING
                                                PUSH RBS-,WB<0>?,              ; CHECK FOR ODD STACK
U 03E8, 0085,473C,0295,84A7,0972,A 376*         NEXT/CM.MFPD-CHECK.SP          ;

                                        .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                        1672:
                                        FREE.1672:
                                                ;0*******FORCE ADDRESS*********;
                                                WRITE M[MDR],SIZE[WORD],       ; PUSH SRC
U 1672, 0881,2592,4010,0DD8,0072,2              CCOP1,NEXT/CO.NOP              ; SET CC'S
```

```
                                ;42567 172A:
                                ;42568 CM.MFPD-CHECK.SP:
                                ;42569          ;0********FORCE ADDRESS*********;
                                ;42570          WRITE M[MDR],SIZE[WORD],        ; PUSH SRC
U 172A, 0881,2592,4010,0DD8,0072,2 ;42571       CCOP1,NEXT/CO.NOP               ; SET CC'S
                                ;42572
                                ;42573 172B:    ;1********FORCE ADDRESS*********;
U 172B, 0C80,0036,4030,0047,00FD,9 ;42574       NEXT/IE.CM.ODD                  ; CAUSE ODD ADDRESS TRAP
                                ;42575
                                ;42576 1668:
                                ;42577 CM.MFPD-TEST.SP:
                                ;42578          ;0********FORCE ADDRESS*********;
                                ;42579          VA_R6_RB-CONX(2) RNUM.EQ.7?,    ; DEC STACK BEFORE PUSHING
                                ;42580          PUSH RBS-,                      ; SAVE STATE
U 1668, 0085,473C,0B95,84A7,0146,C ;42581       NEXT/CM.MFPD-TEST.MODE          ;
                                ;42582 1669:
                                ;42583          ;1********FORCE ADDRESS*********;
U 1669, 0C80,0036,4030,0047,00FD,9 ;42584       NEXT/IE.CM.ODD                  ; CAUSE AN ODD ADDRESS TRAP
```

```
                                ;42585 .TOC     ""      Compatibiiity Mode              : MTPI, MTPD""
                                ;42586
                                ;42587 ;*********************************************************************************
                                ;42588 ;       MTPx    dst.ww
                                ;42589 ;          Input (dst is memory)        VA      Address of dst
                                ;42590 ;
                                ;42591 ;          Input (dst is register)      RNUM    # of register to WRITE
                                ;42592 ;*********************************************************************************
                                ;42593 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42594 CM.MTPD:
                                ;42595 CM.MTPI:
                                ;42596          ;-------------------------------;
                                ;42597          VA_R[R6] Q_M[VA],               ; GET ADDRESS OF STACK
U 03E9, 0481,B004,72B5,84A7,014C,A ;42598       WB<0>?                          ; CHECK FOR ODD STACK ADDRESS
                                ;42599
                                ;42600 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;42601 14CA:    ;0********FORCE ADDRESS*********;
                                ;42602          READ,                           ; READ STACK
                                ;42603          R[R6]_RB+CONX(2),PUSH RBS+,     ; POP SP AND TEST FOR REG MODE PC
U 14CA, 0C85,573D,04D5,8050,0149,4 ;42604       FLAG3?,NEXT/CM.MTPI-TEST.REGMOD ; TEST FOR REGISTER MODE
                                ;42605 14CB:
                                ;42606          ;1********FORCE ADDRESS*********;
U 14CB, 0C80,0036,4030,0047,00FD,9 ;42607       NEXT/IE.CM.ODD                  ; CAUSE ODD ADDRESS TRAP
                                ;42608 =0
                                ;42609 CM.MTPI-TEST.REGMOD:
                                ;42610          ;0-------------------------------;
                                ;42611          VA_Q,                           ; RESTORE VA
U 1494, 0C80,003A,403D,84A7,0147,0 ;42612       NEXT/CM.MTP-WRITE               ;
                                ;42613
                                ;42614          ;1-------------------------------;
U 1495, 0C80,0036,4BBC,C047,0147,0 ;42615       RNUM.EQ.7?                      ; TEST FOR POSSIBLY WRITTING PC
                                ;42616 =00
                                ;42617 CM.MTP-WRITE:
                                ;42618          ;00------------------------------; NOT REG MODE PC
                                ;42619          R[DST.R].SIZ_M[MDR],SIZE[WORD], ; WRITE REGISTER IF REG MODE
                                ;42620          WRITE NOTREG,CCOP1,             ; WRITE MEMORY IF MEM MODE  SET CC'S
U 1470, 0083,2592,401C,4DDA,0072,2 ;42621       NEXT/CO.NOP                     ;
                                ;42622
                                ;42623 =10      ;10------------------------------; REG MODE PC
                                ;42624          PC_M[MDR],                      ; WRITE PC
                                ;42625          CCOP1,SIZE[WORD],               ; SET CC'S
U 1472, 0881,2002,429D,8C87,0149,6 ;42626       WB<0>?,NEXT/CM.TEST.PC          ; TEST FOR ODD ADDRESS
                                ;42627
                                ;42628          ;11------------------------------; NOT REG MODE PC
                                ;42629          R[DST.R].SIZ_M[MDR],SIZE[WORD], ; WRITE REGISTER IF REG MODE
                                ;42630          WRITE NOTREG,CCOP1,             ; WRITE MEMORY IF MEM MODE  SET CC'S
U 1473, 0083,2592,401C,4DDA,0072,2 ;42631       NEXT/CO.NOP                     ;
```

```
                                ;42632 .TOC     ""      Compatibility Mode               : RTI, RTT''
                                ;42633
                                ;42634 ;*****************************************************************************
                                ;42635 ;       RTI
                                ;42636 ;       RTT
                                ;42637 ;*****************************************************************************
                                ;42638 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42639 =1111
                                ;42640 CM.RTI:
                                ;42641 CM.RTT:
                                ;42642          ;1111-------------------------;
                                ;42643          M[TEMP1]_R[R6],MDR_ZEXT(OSR),   ; GET ADDRESS OF STACK
                                ;42644          SET FLAG2,WB<0>?,               ; CHECK FOR ODD STACK ADDRESS
U 017F, 0456,15BE,42B5,85E7,014D,0  ;42645      NEXT/CM-TEST.OSR                ; GO CHECK FOR RESERVED OPERANDS
                                ;42646
                                ;42647 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;42648
                                ;42649 CM.RTI.CONTINUE:
                                ;42650          ;-------------------------------;
                                ;42651          READ,                           ; READ STACK
U 14CC, 0C84,073D,0025,8050,014C,D  ;42652      R[R6]_RB+CONX(4)                ; POP STACK
                                ;42653
                                ;42654          ;-------------------------------;
U 14CD, 0481,2016,401D,8487,014C,E  ;42655      PC_ZEXT(M[MDR]),SIZE[WORD]      ; STUFF PC FROM STACK
                                ;42656
                                ;42657          ;-------------------------------;
U 14CE, 0085,23B7,0010,0047,014C,F  ;42658      R[TEMP0]_M[MDR].RR.16           ; GET DATA TO STUFF PSW WITH
                                ;42659
                                ;42660          ;-------------------------------;
                                ;42661          PSW_M[TEMP0].AND.ZLIT0[1F],     ; WRITE CC INFO INTO PSW
U 14CF, 0D80,0C12,0670,F827,0140,B  ;42662      IR<2-0>?                        ; TEST FOR RTT (BIT 2 SET)
```

```
                                ;42663 =011
                                ;42664 CM.RTI-TEST.PC:
                                ;42665          ;011----------------------------; RTI
                                ;42666          WB_M[PC],                       ; DRIVE PC TO WBUS TO TEST
U 140B, 0481,A592,42B0,0047,0149,6  ;42667      WB<0>?,NEXT/CM.TEST.PC          ; TEST FOR ODD ADDRESS
                                ;42668
                                ;42669          ;111----------------------------; RTT
U 140F, 0C86,0036,4030,0087,0166,A  ;42670      M[TEMP0]_PSL                    ; GET CURRENT PSL
                                ;42671
                                ;42672 166A:    ;*********FORCE ADDRESS*********;
                                ;42673          PSL_M[TEMP0].ANDNOT.ZLIT24[80], ; CLEAR TP SO TRAP WON'T OCCUR
U 166A, 0980,0C93,8034,0007,0140,B  ;42674      NEXT/CM.RTI-TEST.PC             ;
                                ;42675 =0
                                ;42676 CM.TEST.PC:
                                ;42677          ;0------------------------------; PC NOT ODD
U 1496, 0080,0036,4130,0047,003F,9  ;42678      IRD1                            ; ALL OK
                                ;42679
                                ;42680 CM.TEST.PC.ODD:
                                ;42681          ;1------------------------------; PC ODD
                                ;42682          WB_PSL,                         ; READ PSL
U 1497, 0C80,0036,46F0,0087,0147,6  ;42683      WB<31-30>?                      ; TEST TP BIT
                                ;42684
                                ;42685 =10      ;10-----------------------------; TP CLEAR  DO ODD ADDRESS TRAP
U 1476, 0C80,0036,4030,0047,00FD,9  ;42686      NEXT/IE.CM.ODD                  ; BACK TO NATIVE MODE
                                ;42687
                                ;42688          ;11-----------------------------; TP SET
U 1477, 0080,0036,4130,0047,003F,9  ;42689      IRD1                            ; LET THE HARDWARE DO THE WORK
                                ;42690
                                ;42691 14D0:
                                ;42692 CM-TEST.OSR:
                                ;42693          ;*********FORCE ADDRESS*********;
                                ;42694          WB_M[MDR].AND.ZLIT0[38],        ;
U 14D0, 0581,2C12,0A31,C047,0149,8  ;42695      WX.EQ.0?,NEXT/CM-TEST.OSR1      ; TEST FOR RESERVED OPERAND
                                ;42696
                                ;42697 14D1:    ;*********FORCE ADDRESS*********;
U 14D1, 0C80,0036,4030,0047,00FD,9  ;42698      NEXT/IE.CM.ODD                  ; CAUSE ODD ADDRESS TRAP
                                ;42699
                                ;42700 =0
                                ;42701 CM-TEST.OSR1:
                                ;42702          ;-------------------------------;
U 1498, 0C80,0036,4C30,0047,0010,F  ;42703      NEXT/IE.CM.RESOP                ; TAKE A RESERVED OPERAND FAULT
                                ;42704
                                ;42705          ;-------------------------------;
                                ;42706          VA_M[TEMP1],                    ; LOAD VA FOR RTS
U 1499, 0880,1002,45BD,84A7,0141,0  ;42707      FLAG<2-0>?,NEXT/CM-TEST.OSR.CONT ; SEE WHERE WE CAME FROM
                                ;42708 =000
                                ;42709 CM-TEST.OSR.CONT:
                                ;42710 =001     ;001----------------------------;
U 1411, 0C80,0036,4030,0047,00FD,A  ;42711      NEXT/IE.CM.FAULT                ; CONTINE FROM IOT OR BPT
                                ;42712
                                ;42713          ;010----------------------------; CONTINUE FROM RTS
                                ;42714          READ,                           ; GET SAVED REGISTER
                                ;42715          R6_RB+CONX(2) RNUM.EQ.7?,       ; POP SP
U 1412, 0C84,073D,0B95,8050,0146,8  ;42716      NEXT/CM.RTS.CONTINUE            ; TEST FOR RET PC
                                ;42717
```

```
                                                                I  2
                                    ;42718 =100     ;100----------------------------; CONTINUE FROM RTI RTT
                                    ;42719          VA_ZEXT(M[TEMP1]),               ;ZERO EXTEND VA INCASE OF UNALIGNED READ
U 1414, 0080,1016,401D,84A7,014C,C  ;42720          SIZE[WORD],NEXT/CM.RTI.CONTINUE ;
                                    ;42721 =
```

```
                                        ;42722 .TOC "  Comp Mode Operand Specifier     : CM.OS.RED"
                                        ;42723
                                        ;42724 ;*****************************************************************************
                                        ;42725 ;       MDR is saved in the Q-register and is always destroyed.
                                        ;42726 ;       Evaluates read type operands and places the result in MDR.
                                        ;42727 ;       MTEMP0 is destroyed.
                                        ;42728 ;       Read access for the operand is checked.
                                        ;42729 ;*****************************************************************************
                                        ;42730
                                        ;42731 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                        ;42732 =0000
                                        ;42733 CM.OS.RED:
                                        ;42734         ;0000-------------------------; Rn        REGISTER MODE
                                        ;42735         Q_M[MDR] MDR_R[GPR.R],        ; PLACE OPERAND (GPR(RNUM)) IN MDR
U 02D0, 0C81,2004,707C,C467,0000,1      ;42736         IRDX [1]                      ; SAVE MDR IN Q BEFORE CLOBBERING IT
                                        ;42737
                                        ;42738         ;0001-------------------------; PC        REGISTER MODE PC
                                        ;42739         Q_M[MDR],                     ; SAVE MDR IN QREG
U 02D1, 0481,2592,5030,0047,014D,2      ;42740         NEXT/CM.OS.RED-REGPC          ;
                                        ;42741
                                        ;42742         ;0010-------------------------; (Rn)      REGISTER DEFERRED MODE
                                        ;42743         VA_R[GPR.R],                  ; ADDRESS OF OPERAND IS PLACED IN VA
                                        ;42744         ODD ADDRESS?,                 ; TEST FOR ODD ADDRESS
U 02D2, 0480,05BE,487C,C4A7,0149,A      ;42745         NEXT/CM.OS.RED-ODD.CHECK1     ;
                                        ;42746
                                        ;42747         ;0011-------------------------; (PC)      REGISTER DEFERRED MODE PC
                                        ;42748         VA_M[PC],                     ; ADDRESS OF OPERAND IS PLACED IN VA
U 02D3, 0481,A002,403D,84A7,0149,A      ;42749         NEXT/CM.OS.RED-ODD.CHECK1     ; ALREADY KNOW IT'S EVEN BUT SAVE A WORD
                                        ;42750
                                        ;42751         ;0100-------------------------; (Rn)+     AUTOINCREMENT MODE
                                        ;42752         Q_M[MDR] VA_R[GPR.R],         ; GET OPERAND ADDRESS FROM REGISTER
                                        ;42753         ODD ADDRESS?,                 ; TEST FOR ODD ADDRESS
U 02D4, 0081,2004,787C,C4A7,0149,C      ;42754         NEXT/CM.OS.RED-ODD.CHECK2     ; ALSO SAVE MDR IN Q
                                        ;42755
                                        ;42756         ;0101-------------------------; (SP)+     AUTOINCREMENT MODE SP
                                        ;42757         Q_M[MDR] VA_R[GPR.R],         ; GET OPERAND ADDRESS FROM REGISTER
                                        ;42758         ODD ADDRESS?,                 ; TEST FOR ODD ADDRESS
U 02D5, 0881,2004,787C,C4A7,0149,E      ;42759         NEXT/CM.OS.RED-ODD.CHECK3     ; ALSO SAVE MDR IN Q
                                        ;42760
                                        ;42761         ;0110-------------------------; @(Rn)+    AUTO-INC DEFERRED MODE
                                        ;42762         Q_M[MDR] VA_R[GPR.R],         ; GET ADDRESS OF ADDRESS OF OPERAND
                                        ;42763         WB<0>?,                       ; TEST FOR ODD ADDRESS
U 02D6, 0881,2004,72BC,C4A7,014A,E      ;42764         NEXT/CM.OSR-GET.IND.1         ; FINNISH IN THE COMMON ROUTINE
                                        ;42765
                                        ;42766         ;0111-------------------------; @#Addr    AUTO-INC DEFERRED MODE PC
                                        ;42767         VA_ZEXT(XB) PC_PC+2,          ; GET ABSOLUTE ADDRESS FROM I-STREAM
                                        ;42768         ODD ADDRESS?,                 ; TEST FOR ODD ADDRESS
U 02D7, 0C81,7016,485D,A4A7,0149,A      ;42769         NEXT/CM.OS.RED-ODD.CHECK1     ;
                                        ;42770
                                        ;42771         ;1000-------------------------; -(Rn)     AUTO-DECREMENT MODE
                                        ;42772         D_CONX.SIZ,SIZE[IDEP],        ; GET AMOUNT TO DECREMENT BY
                                        ;42773         PUSH RBS-,R[GPR.R],           ; SAVE REGISTER STATE
U 02D8, 0C81,4737,203C,C047,014D,6      ;42774         NEXT/CM.OS.RED-AUTO.DEC       ;
```

```
                                ;42775          ;1001----------------------------; -(SP)     AUTO-DECREMENT MODE SP
                                ;42776          VA_R[GPR.R].SIZ_RB-CONX(2),      ; OPERAND ADDRESS IS GPR(RUNM) - SIZE
                                ;42777          PUSH RBS-,                       ; SAVE REG CONTENTS FOR LATER
                                ;42778          ODD ADDRESS?,                    ; TEST FOR ODD ADDRESS
U 02D9, 0C83,473C,085C,C4A7,0949,A 379*  ;42779 NEXT/CM.OS.RED-ODD.CHECK1         ;
                                ;42780
                                ;42781          ;1010----------------------------; @-(Rn)    AUTO-DEC DEFERRED MODE
                                ;42782          VA_R[GPR.R].SIZ_RB-CONX(2),      ; OPERAND ADDRESS IS GPR(RUNM) - SIZE
                                ;42783          PUSH RBS-,                       ; SAVE REG CONTENTS FOR LATER
                                ;42784          RNUM.EQ.7?,                      ; TEST FOR SPECIAL CASE
U 02DA, 0483,473C,0B9C,C4A7,0147,8       ;42785 NEXT/CM.OSR-GET.IND.2            ; FINNISH IN THE COMMON ROUTINE
                                ;42786
                                ;42787          ;1011----------------------------; -(PC)     AUTO-DECREMENT MODE PC
                                ;42788          VA_M[PC]-ZLIT0[2],               ; AUTO-DEC PC BY 2 ALWAYS
U 02DB, 0D81,AC10,0030,14A7,014D,4       ;42789 NEXT/CM.OS.RED-ADEC.PC           ;
                                ;42790
                                ;42791          ;1100----------------------------; X(Rn)     INDEX MODE
                                ;42792          VA_SEXT(XB)+R[GPR.R] PC_PC+2,    ; OPERAND ADDRESS IS REG + DISP
                                ;42793          ODD ADDRESS?,                    ; CHECK FOR ODD ADDRESS
U 02DC, 0481,7815,085C,E4A7,0949,A 352*  ;42794 NEXT/CM.OS.RED-ODD.CHECK1         ;
                                ;42795
                                ;42796          ;1101----------------------------; Addr X(PC)    RELATIVE MODE
                                ;42797          VA_PC+SEXT(XB)+2 PC_PC+2,        ; ADDRESS OF OPERAND IS PC + DISP
                                ;42798          ODD ADDRESS?,                    ; CHECK FOR ODD ADDRESS
U 02DD, 0881,7816,485D,A407,0149,A       ;42799 NEXT/CM.OS.RED-ODD.CHECK1         ;
                                ;42800
                                ;42801          ;1110----------------------------; @ADDR @X(Rn)    INDEX DEFERRED MODE
                                ;42802          WB_R[GPR.R],                     ; MUST DO THIS FOR SPASTA
                                ;42803          RNUM.EQ.7?,                      ; TEST FOR REGISTER MODE
U 02DE, 0480,05BE,4BBC,C047,0148,4       ;42804 NEXT/CM.OSR-GET.IND.3            ; FINNISH IN THE COMMON ROUTINE
                                ;42805
                                ;42806 CM.OS.RED-IMED:
                                ;42807          ;1111----------------------------; #CONS     IMMEDIATE MODE
                                ;42808          Q_M[MDR],                        ; SAVE MDR BEFORE CLOBBERING IT
U 02DF, 0C81,2592,5030,0047,014D,5       ;42809 NEXT/CM.OS.RED-IMMEDIATE         ;
                                ;42810
                                ;42811 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                ;42812 CM.OS.RED-REGPC:
                                ;42813          ;--------------------------------;
                                ;42814          MDR_M[PC],                       ; PLACE OPERAND PC INTO MDR
U 14D2, 0881,A002,407D,8467,0000,1       ;42815 IRDX [1]                         ;
                                ;42816 =0
                                ;42817 CM.OS.RED-ODD.CHECK1:
                                ;42818          ;0--------------------------------; ADDRESS OK
                                ;42819          READ,SIZE[IDEP],                 ; GET OPERAND INTO MDR
                                ;42820          Q_M[MDR],                        ; SAVE OLD MDR BEFORE KILLING IT
U 149A, 0881,2592,5070,0050,0000,1       ;42821 IRDX [1]                         ;
                                ;42822
                                ;42823          ;1--------------------------------; ODD ADDRESS
U 149B, 0C80,0036,4030,0047,00FD,9       ;42824 NEXT/IE.CM.ODD                   ; BAD NEWS
```

```
                                ;42825 =0
                                ;42826 CM.OS.RED-ODD.CHECK2:
                                ;42827          :0------------------------------; ADDRESS OK
                                ;42828          READ,                            ; GET OPERAND FROM MEMORY
                                ;42829          D_R[GPR.R]+CONX.SIZ,PUSH RBS+,   ; UPDATE REGISTER INTO DREG
                                ;42830          SIZE[IDEP],                      ; SIZE DETERMINED BY INSTRUCTION
U 149C, 0881,573D,203C,C050,014D,3  ;42831      NEXT/CM.OS.RED-ODD.CHECK2A       ;
                                ;42832
                                ;42833          ;1------------------------------; ODD ADDRESS
U 149D, 0C80,0036,4030,0047,00FD,9  ;42834      NEXT/IE.CM.ODD                   ; BAD NEWS
                                ;42835
                                ;42836 CM.OS.RED-ODD.CHECK2A:
                                ;42837          ;--------------------------------;
                                ;42838          R[GPR.R].SIZ_D,                  ; AUTO-INC GPR REGISTER
                                ;42839          SIZE[WORD],                      ; BE SURE TO WRITE ALL 16 BITS
U 14D3, 0882,05B2,405C,C047,0000,1  ;42840      IRDX [1]                         ;
                                ;42841 =0
                                ;42842 CM.OS.RED-ODD.CHECK3:
                                ;42843          :0------------------------------; ADDRESS OK
                                ;42844          READ,                            ; GET OPERAND INTO MDR
                                ;42845          R[GPR.R].SIZ_RB+CONX(2),         ; UPDATE REGISTER BY TWO
                                ;42846          PUSH RBS+,                       ; SAVE STATE IN CASE OF ERRORS
U 149E, 0883,573D,005C,C050,0000,1  ;42847      IRDX [1]                         ;
                                ;42848
                                ;42849          ;1------------------------------; ODD ADDRESS
U 149F, 0C80,0036,4030,0047,00FD,9  ;42850      NEXT/IE.CM.ODD                   ; BAD NEWS
                                ;42851
                                ;42852 CM.OS.RED-ADEC.PC:
                                ;42853          ;--------------------------------;
                                ;42854          PC_M[VA],                        ; AUTODEC THE PC
U 14D4, 0881,B002,403D,8487,0149,A  ;42855      NEXT/CM.OS.RED-ODD.CHECK1        ;
                                ;42856
                                ;42857 CM.OS.RED-IMMEDIATE:
                                ;42858          ;--------------------------------;
                                ;42859          MDR_XB PC_PC+2,                  ; GET IMMEDIATE DATA FROM XB
U 14D5, 0C81,7002,401D,A467,014C,0  ;42860      NEXT/CM.IRDX                     ; LET PC SETTLE
                                ;42861
                                ;42862 CM.OS.RED-AUTO.DEC:
                                ;42863          ;--------------------------------;
                                ;42864          VA_R[GPR.R].SIZ_RB-D,            ; OPERAND ADDRESS IS GPR(RUNM) - SIZE
                                ;42865          SIZE[WORD],ODD ADDRESS?,         ; TEST FOR ODD ADDRESS
U 14D6, 0C82,0023,085C,C4A7,0949,A 347*  ;42866 NEXT/CM.OS.RED-ODD.CHECK1         ;
```

```
                                ;42867 .TOC "  Comp Mode Operand Specifier     : CM.OS.MOD"
                                ;42868
                                ;42869
                                ;42870 ;*************************************************************************
                                ;42871 ;       MDR is saved in the Q-register and is always destroyed.
                                ;42872 ;       Evaluates modify type operands and places the result in MDR.
                                ;42873 ;       MTEMP0 is destroyed.
                                ;42874 ;       Write access for one byte is checked
                                ;42875 ;*************************************************************************
                                ;42876
                                ;42877 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;42878 =0000
                                ;42879 CM.OS.MOD:
                                ;42880          ;0000-------------------------; Rn        REGISTER MODE
                                ;42881          Q_M[MDR] MDR_R[GPR.R],        ; PLACE OPERAND (GPR(RNUM)) IN MDR
                                ;42882          SET FLAG2,                    ; TO TELL XOR ABOUT REGISTER MODE
U 02E0, 0451,2004,707C,C467,0000,1              ;42883          IRDX [1]                      ; SAVE MDR IN Q BEFORE CLOBBERING IT
                                ;42884
                                ;42885          ;0001-------------------------; PC        REGISTER MODE PC
                                 42886          Q_M[MDR],                     ; SAVE MDR IN QREG
                                ;42887          SET FLAG2,                    ; TO TELL XOR ABOUT REGISTER MODE
U 02E1, 0C51,2592,5030,0047,014D,2              ;42888          NEXT/CM.OS.RED-REGPC          ;
                                ;42889
                                ;42890          ;0010-------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;42891          VA_R[GPR.R],                  ; ADDRESS OF OPERAND IS PLACED IN VA
                                ;42892          ODD ADDRESS?,                 ; TEST FOR ODD ADDRESS
U 02E2, 0480,05BE,487C,C4A7,014A,0              ;42893          NEXT/CM.OS.MOD-ODD.CHECK1     ;
                                ;42894
                                ;42895          ;0011-------------------------; (PC)      REGISTER DEFERRED MODE PC
                                ;42896          VA_M[PC],                     ; ADDRESS OF OPERAND IS PLACED IN VA
U 02E3, 0481,A002,403D,84A7,014A,0              ;42897          NEXT/CM.OS.MOD-ODD.CHECK1     ; ALREADY KNOW IT'S EVEN BUT SAVE A WORD
                                ;42898
                                ;42899          ;0100-------------------------; (Rn)+     AUTOINCREMENT MODE
                                ;42900          Q_M[MDR] VA_R[GPR.R],         ; GET OPERAND ADDRESS FROM REGISTER
                                ;42901          ODD ADDRESS?,                 ; TEST FOR ODD ADDRESS
U 02E4, 0881,2004,787C,C4A7,014A,2              ;42902          NEXT/CM.OS.MOD-ODD.CHECK2     ; ALSO SAVE MDR IN Q
                                ;42903
                                ;42904          ;0101-------------------------; (SP)+     AUTOINCREMENT MODE SP
                                ;42905          Q_M[MDR] VA_R[GPR.R],         ; GET OPERAND ADDRESS FROM REGISTER
                                ;42906          ODD ADDRESS?,                 ; TEST FOR ODD ADDRESS
U 02E5, 0881,2004,787C,C4A7,014A,4              ;42907          NEXT/CM.OS.MOD-ODD.CHECK3     ; ALSO SAVE MDR IN Q
                                ;42908
                                ;42909          ;0110-------------------------; @(Rn)+    AUTO-INC DEFERRED MODE
                                ;42910          Q_M[MDR] VA_R[GPR.R],         ; GET ADDRESS OF ADDRESS OF OPERAND
                                ;42911          SET FLAG0,                    ; INFO FOR COMMON ROUTINE
                                ;42912          WE<0>?,                       ; TEST FOR ODD ADDRESS
U 02E6, 0841,2004,72BC,C4A7,014A,E              ;42913          NEXT/CM.OSR-GET.IND.1         ; FINNISH IN THE COMMON ROUTINE
                                ;42914
                                ;42915          ;0111-------------------------; @#Addr    AUTO-INC DEFERRED MODE PC
                                ;42916          VA_ZEXT(XB) PC_PC+2,          ; GET ABSOLUTE ADDRESS FROM I-STREAM
                                ;42917          ODD ADDRESS?,                 ; TEST FOR ODD ADDRESS
U 02E7, 0C81,7016,485D,A4A7,014A,0              ;42918          NEXT/CM.OS.MOD-ODD.CHECK1     ;
```

```
                                    ;42919          ;1000--------------------------; -(Rn)     AUTO-DECREMENT MODE
                                    ;42920          D_CONX.SIZ,SIZE[IDEP],         ; GET SIZE TO DEC REGISTER BY
                                    ;42921          PUSH RBS-,R[GPR.R],            ; SAVE REGISTER STATE
U 02E8, 0C81,4737,203C,C047,014D,9  ;42922          NEXT/CM.OS.MOD-AUTO.DEC        ;
                                    ;42923
                                    ;42924          ;1001--------------------------; -(SP)     AUTO-DECREMENT MODE SP
                                    ;42925          VA_R[GPR.R].SIZ_RB-CONX(2),    ; OPERAND ADDRESS IS GPR(RUNM) - SIZE
                                    ;42926          PUSH RBS-,                     ; SAVE REG CONTENTS FOR LATER
                                    ;42927          ODD ADDRESS?,                  ; TEST FOR ODD ADDRESS
U 02E9, 0C83,473C,085C,C4A7,094A,0 379* ;42928      NEXT/CM.OS.MOD-ODD.CHECK1      ;
                                    ;42929
                                    ;42930          ;1010--------------------------; @-(Rn)    AUTO-DEC DEFERRED MODE
                                    ;42931          VA_R[GPR.R].SIZ_RB-CONX(2),    ; OPERAND ADDRESS IS GPR(RUNM) - SIZE
                                    ;42932          PUSH RBS-,                     ; SAVE REG CONTENTS FOR LATER
                                    ;42933          SET FLAG0,                     ; INFO FOR COMMON ROUTINE
                                    ;42934          RNUM.EQ.7?,                    ; TEST FOR SPECIAL CASE
U 02EA, 0443,473C,0B9C,C4A7,0147,8  ;42935          NEXT/CM.OSR-GET.IND.2          ; FINNISH IN THE COMMON ROUTINE
                                    ;42936
                                    ;42937          ;1011--------------------------; -(PC)     AUTO-DECREMENT MODE PC
                                    ;42938          VA_M[PC]-ZLIT0[2],             ; AUTO-DEC PC BY 2 ALWAYS
U 02EB, 0D81,AC10,0030,14A7,014D,8  ;42939          NEXT/CM.OS.MOD-ADEC.PC         ;
                                    ;42940
                                    ;42941          ;1100--------------------------; X(Rn)     INDEX MODE
                                    ;42942          VA_SEXT(XB)+R[GPR.R] PC_PC+2,  ; OPERAND ADDRESS IS REG + DISP
                                    ;42943          ODD ADDRESS?,                  ; CHECK FOR ODD ADDRESS
U 02EC, 0481,7815,085C,E4A7,094A,0 352* ;42944      NEXT/CM.OS.MOD-ODD.CHECK1      ;
                                    ;42945
                                    ;42946          ;1101--------------------------; Addr X(PC)    RELATIVE MODE
                                    ;42947          VA_PC+SEXT(XB)+2 PC_PC+2,      ; ADDRESS OF OPERAND IS PC + DISP
                                    ;42948          ODD ADDRESS?,                  ; CHECK FOR ODD ADDRESS
U 02ED, 0881,7816,485D,A407,014A,0  ;42949          NEXT/CM.OS.MOD-ODD.CHECK1      ;
                                    ;42950
                                    ;42951          ;1110--------------------------; @ADDR @X(Rn)    INDEX DEFERRED MODE
                                    ;42952          WB_R[GPR.R],                   ; MUST DO THIS FOR SPASTA
                                    ;42953          SET FLAG0,                     ; INFO FOR COMMON ROUTINE
                                    ;42954          RNUM.EQ.7?,                    ; TEST FOR REGISTER MODE
U 02EE, 0440,05BC,4BBC,C047,0148,4  ;42955          NEXT/CM.OSR-GET.IND.3          ; FINNISH IN THE COMMON ROUTINE
                                    ;42956
                                    ;42957          ;1111--------------------------; #CONS     IMMEDIATE MODE
                                    ;42958          VA_M[PC],                      ; GET ADDRESS OF IMMEDIATE DATA
U 02EF, 0481,A002,403D,84A7,002D,F  ;42959          NEXT/CM.OS.RED-IMED            ;
                                    ;42960
                                    ;42961 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                    ;42962 =0
                                    ;42963 CM.OS.MOD-ODD.CHECK1:
                                    ;42964          ;0--------------------------; ADDRESS OF
                                    ;42965          READ.MOD,SIZE[IDEP],           ; GET OPERAND [ ] MDR
                                    ;42966          Q_M[MDR],                      ; SAVE OLD MD[ ] BEFORE KILLING IT
U 14A0, 0081,2592,5070,0054,0000,1  ;42967          IRDX [1]                       ;
                                    ;42968
                                    ;42969          ;1--------------------------; ODD ADDRESS
U 14A1, 0C80,0036,4030,0047,00FD,9  ;42970          NEXT/IE.CM.ODD                 ; BAD NEWS
```

```
                                ;42971 =0
                                ;42972 CM.OS.MOD-ODD.CHECK2:
                                ;42973          ;0------------------------------; ADDRESS OK
                                ;42974          READ.MOD,                       ; READ OPERAND FROM MEMORY
                                ;42975          D R[GPR.R]+CONX.SIZ,            ; INC REGISTER AND SAVE IN D
                                ;42976          PUSH RBS+,SIZE[IDEP],           ; SAVE STATE OF REGISTER
U 14A2, 0881,573D,203C,C054,014D,7 ;42977       NEXT/CM.OS.MOD-ODD.CHECK2A      ;
                                ;42978
                                ;42979          ;1------------------------------; ODD ADDRESS
U 14A3, 0C80,0036,4030,0047,00FD,9 ;42980       NEXT/IE.CM.ODD                  ; BAD NEWS
                                ;42981
                                ;42982 CM.OS.MOD-ODD.CHECK2A:
                                ;42983          ;------------------------------;
                                ;42984          R[GPR.R].SIZ_D,                 ; UPDATE REGISTER
                                ;42985          SIZE[WORD],                     ; WRITE ALL 16 BITS
U 14D7, 0882,05B2,405C,C047,0000,1 ;42986       IRDX [1]                        ;
                                ;42987 =0
                                ;42988 CM.OS.MOD-ODD.CHECK3:
                                ;42989          ;0------------------------------; ADDRESS OK
                                ;42990          READ.MOD,                       ; GET OPERAND INTO MDR
                                ;42991          R[GPR.R].SIZ_RB+CONX(2),        ; UPDATE REGISTER BY TWO
                                ;42992          PUSH RBS+,                      ; SAVE STATE IN CASE OF ERRORS
U 14A4, 0083,573D,005C,C054,0000,1 ;42993       IRDX [1]                        ;
                                ;42994
                                ;42995          ;1------------------------------; ODD ADDRESS
U 14A5, 0C80,0036,4030,0047,00FD,9 ;42996       NEXT/IE.CM.ODD                  ; BAD NEWS
                                ;42997
                                ;42998 CM.OS.MOD-ADEC.PC:
                                ;42999          ;------------------------------;
                                ;43000          PC_M[VA],                       ; AUTODEC THE PC
U 14D8, 0881,B002,403D,8487,0149,A ;43001       NEXT/CM.OS.RED-ODD.CHECK1       ;
                                ;43002
                                ;43003 CM.OS.MOD-AUTO.DEC:
                                ;43004          ;------------------------------;
                                ;43005          VA_R[GPR.R].SIZ_RB-D,           ; OPERAND ADDRESS IS GPR(RUNM) - SIZE
                                ;43006          SIZE[WORD],ODD ADDRESS?,        ; TEST FOR ODD ADDRESS
U 14D9, 0C82,0023,085C,C4A7,094A,0 347* ;43007  NEXT/CM.OS.MOD-ODD.CHECK1       ;
```

```
                                ;43008 .TOC "  Comp mode Operand Specifier     : CM.OS.WRT"
                                ;43009
                                ;43010
                                ;43011 ;*****************************************************************************
                                ;43012 ;       MDR is saved in the Q-register and is always destroyed.
                                ;43013 ;       Evaluates write and address type operands and places the result in MDR.
                                ;43014 ;       MTEMP0 also gets the result
                                ;43015 ;       No access is checked
                                ;43016 ;*****************************************************************************
                                ;43017
                                ;43018 .REGION/IRDX.R1L,IRDX.R1H/IRDX.R2L,IRDX.R2H
                                ;43019 =0000
                                ;43020 CM.OS.WRT:
                                ;43021         ;0000-----------------------------; Rn        REGISTER MODE
                                ;43022         Q_M[MDR],                        ; SAVE MDR IN Q TO BE CONSISTANT
                                ;43023         SET FLAG3,                       ; SET FLAG3 TO INDICATE REGISTER MODE
U 02F0, 0859,2592,5070,0047,0000,1      ;43024         IRDX [1]                         ;
                                ;43025
                                ;43026         ;0001-----------------------------; PC        REGISTER MODE PC
                                ;43027         Q_M[MDR],                        ; SAVE MDR IN Q TO BE CONSISTANT
                                ;43028         SET FLAG3,                       ; SET FLAG3 TO INDICATE REGISTER MODE
U 02F1, 0859,2592,5070,0047,0000,1      ;43029         IRDX [1]                         ;
                                ;43030
                                ;43031         ;0010-----------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;43032         VA_M[TEMP0]_R[GPR.R],            ; ADDRESS OF OPERAND IS PLACED IN VA
                                ;43033         ODD ADDRESS?,                    ; TEST FOR ODD ADDRESS
U 02F2, 0486,05BE,487C,C4A7,014A,6      ;43034         NEXT/CM.OS.WRT-ODD.CHECK1        ;
                                ;43035
                                ;43036         ;0011-----------------------------; (PC)      REGISTER DEFERRED MODE PC
                                ;43037         VA_MTEMP0_PC,                    ; ADDRESS OF OPERAND IS PLACED IN VA
U 02F3, 0487,A002,403D,84A7,014A,6      ;43038         NEXT/CM.OS.WRT-ODD.CHECK1        ; ALREADY KNOW IT'S EVEN BUT SAVE A WORD
                                ;43039
                                ;43040         ;0100-----------------------------; (Rn)+     AUTOINCREMENT MODE
                                ;43041         Q_MDR VA_MTEMP0_R[GPR.R],        ; GET OPERAND ADDRESS FROM REGISTER
                                ;43042         ODD ADDRESS?,                    ; TEST FOR ODD ADDRESS
U 02F4, 0887,2004,787C,C4A7,014A,8      ;43043         NEXT/CM.OS.WRT-ODD.CHECK2        ; ALSO SAVE MDR IN Q
                                ;43044
                                ;43045         ;0101-----------------------------; (SP)+     AUTOINCREMENT MODE SP
                                ;43046         Q_MDR VA_MTEMP0_R[GPR.R],        ; GET OPERAND ADDRESS FROM REGISTER
                                ;43047         ODD ADDRESS?,                    ; TEST FOR ODD ADDRESS
U 02F5, 0087,2004,787C,C4A7,014A,A      ;43048         NEXT/CM.OS.WRT-ODD.CHECK3        ; ALSO SAVE MDR IN Q
                                ;43049
                                ;43050         ;0110-----------------------------; @(Rn)+    AUTO-INC DEFERRED MODE
                                ;43051         Q_MDR VA_MTEMP0_R[GPR.R],        ; GET OPERAND ADDRESS FROM REGISTER
                                ;43052         SET FLAG7,                       ; INFO FOR COMMON ROUTINE
                                ;43053         WB<C>?,                          ; TEST FOR ODD ADDRESS
U 02F6, 004F,2004,72BC,C4A7,014A,E      ;43054         NEXT/CM.OSR-GET.IND.1            ; FINNISH IN THE COMMON ROUTINE
                                ;43055
                                ;43056         ;0111-----------------------------; @#Addr    AUTO-INC DEFERRED MODE PC
                                ;43057         VA_MTEMP0_ZEXT(XB) PC_PC+2,      ; GET ABSOLUTE ADDRESS FROM I-STREAM
                                ;43058         ODD ADDRESS?,                    ; TEST FOR ODD ADDRESS
U 02F7, 0C87,7016,485D,A4A7,014A,6      ;43059         NEXT/CM.OS.WRT-ODD.CHECK1        ;
```

```
                                    ;43060          ;1000--------------------------; -(Rn)      AUTO-DECREMENT MODE
                                    ;43061          D_CONX.SIZ,SIZE[IDEP],          ; SAVE SIZE TO DEC REGISTER BY
                                    ;43062          PUSH RBS-,R[GPR.R],             ; SAVE REGISTER STATE
U 02F8, 0481,4737,203C,C047,014D,E  ;43063          NEXT/CM.OS.WRT-AUTO.DEC         ;
                                    ;43064
                                    ;43065          ;1001--------------------------; -(SP)      AUTO-DECREMENT MODE SP
                                    ;43066          R[GPR.R].SIZ_RB-CONX(2),        ; OPERAND ADDRESS IS GPR(RUNM) - SIZE
                                    ;43067          PUSH RBS-,                      ; SAVE REG CONTENTS FOR LATER
                                    ;43068          ODD ADDRESS?,                   ; TEST FOR ODD ADDRESS
U 02F9, 0C83,473C,085C,C047,094A,C 379* ;43069      NEXT/CM.OS.WRT-ODD.CHECK4       ;
                                    ;43070
                                    ;43071          ;1010--------------------------; @-(Rn)     AUTO-DEC DEFERRED MODE
                                    ;43072          VA_R[GPR.R].SIZ_RB-CONX(2),     ; OPERAND ADDRESS IS GPR(RUNM) - SIZE
                                    ;43073          PUSH RBS-,                      ; SAVE REG CONTENTS FOR LATER
                                    ;43074          SET FLAG1,                      ; INFO FOR COMMON ROUTINE
                                    ;43075          RNUM.EQ.7?,                     ; TEST FOR SPECIAL CASE
U 02FA, 0C4B,473C,0B9C,C4A7,0147,8  ;43076          NEXT/CM.OSR-GET.IND.2           ; FINNISH IN THE COMMON ROUTINE
                                    ;43077
                                    ;43078          ;1011--------------------------; -(PC)      AUTO-DECREMENT MODE PC
                                    ;43079          PC_M[PC]-ZLIT0[2],              ; DEC PC
U 02FB, 0D81,AC10,0030,1487,014D,C  ;43080          NEXT/CM.OS.WRT-ADEC.PC          ;
                                    ;43081
                                    ;43082          ;1100--------------------------; X(Rn)      INDEX MODE
                                    ;43083          VA_MTEMP0_SEXT(XB)+R[GPR.R] PC_PC+2,; OPERAND ADDRESS IS REG + DISP
U 02FC, 0487,7815,001C,E4A7,014D,A  ;43084          NEXT/CM.OS.WRT-ODD.CHECK1A      ;
                                    ;43085
                                    ;43086          ;1101--------------------------; Addr X(PC)    RELATIVE MODE
                                    ;43087          VA_MTEMP0_PC+SEXT(XB)+2 PC_PC+2,; OPERAND ADDRESS IS REG + DISP
U 02FD, 0087,7816,401D,A407,014D,A  ;43088          NEXT/CM.OS.WRT-ODD.CHECK1A      ;
                                    ;43089
                                    ;43090          ;1110--------------------------; @ADDR @X(Rn)    INDEX DEFERRED MODE
                                    ;43091          WB_R[GPR.R],                    ; MUST DO THIS FOR SPASTA
                                    ;43092          SET FLAG1,                      ; INFO FOR COMMON ROUTINE
                                    ;43093          RNUM.EQ.7?,                     ; TEST FOR REGISTER MODE
U 02FE, 0C48,05BE,4BBC,C047,0148,4  ;43094          NEXT/CM.OSR-GET.IND.3           ; FINNISH IN THE COMMON ROUTINE
                                    ;43095
                                    ;43096          ;1111--------------------------; #CONS     IMMEDIATE MODE
                                    ;43097          VA_MTEMP0_PC,                   ; GET ACTUAL ADDRESS
U 02FF, 0487,A002,403D,84A7,014D,D  ;43098          NEXT/CM.OS.WRT-IMED             ;
                                    ;43099
                                    ;43100 .REGION/CMODE.R1L,CMODE.R1H/CMODE.R2L,CMODE.R2H/CMODE.R3L,CMODE.R3H
                                    ;43101 CM.OS.WRT-ODD.CHECK1A:
                                    ;43102          ;-------------------------------;
                                    ;43103          R[TEMP0]_M[VA],                 ; PUT VA IN TEMP0
U 14DA, 0485,B592,4870,0047,014A,6  ;43104          ODD ADDRESS?                    ; TEST FOR ODD ADDRESS
```

```
                                ;43105 =0
                                ;43106 CM.OS.WRT-ODD.CHECK1:
                                ;43107          ;0------------------------------; ADDRESS OK
                                ;43108          Q_M[MDR],                       ; SAVE OLD MDR BEFORE KILLING IT
U 14A6, 0881,2592,5070,0047,0000,1 ;43109       IRDX [1]                        ;
                                ;43110
                                ;43111          ;1------------------------------; ODD ADDRESS
U 14A7, 0C80,0036,4030,0047,00FD,9 ;43112       NEXT/IE.CM.ODD                  ; BAD NEWS
                                ;43113 =0
                                ;43114 CM.OS.WRT-ODD.CHECK2:
                                ;43115          ;0------------------------------; ADDRESS OK
                                ;43116          D_CONX.SIZ,SIZE[IDEP],          ; GET SIZE TO ADD TO REGISTER
                                ;43117          PUSH RBS+,R[GPR.R],             ; SAVE REGISTER STATE
U 14A8, 0081,5737,203C,C047,014D,8 ;43118       NEXT/CM.OS.WRT-ODD.CHECK2A      ;
                                ;43119
                                ;43120          ;1------------------------------; ODD ADDRESS
U 14A9, 0C80,0036,4030,0047,00FD,9 ;43121       NEXT/IE.CM.ODD                  ; BAD NEWS
                                ;43122
                                ;43123 CM.OS.WRT-ODD.CHECK2A:
                                ;43124          ;------------------------------;
                                ;43125          R[GPR.R].SIZ_RB+D,              ; UPDATE REGISTER
U 14DB, 0C82,0021,005C,C047,0000,1 ;43126       SIZE[WORD],IRDX [1]             ;
                                ;43127 =0
                                ;43128 CM.OS.WRT-ODD.CHECK3:
                                ;43129          ;0------------------------------; ADDRESS OK
                                ;43130          R[GPR.R].SIZ_RB+CONX(2),        ; UPDATE REGISTER BY TWO
                                ;43131          PUSH RBS+,                      ; SAVE STATE IN CASE OF ERRORS
U 14AA, 0883,573D,005C,C047,0000,1 ;43132       IRDX [1]                        ;
                                ;43133
                                ;43134          ;1------------------------------; ODD ADDRESS
U 14AB, 0C80,0036,4030,0047,00FD,9 ;43135       NEXT/IE.CM.ODD                  ; BAD NEWS
                                ;43136 =0
                                ;43137 CM.OS.WRT-ODD.CHECK4:
                                ;43138          ;0------------------------------; ADDRESS OK
                                ;43139          Q_MDR_VA_MTEMP0_R[GPR.R],       ; SAVE MDR AND LOAD VA AND TEMP0
U 14AC, 0C87,2004,707C,C4A7,0000,1 ;43140       IRDX [1]                        ;
                                ;43141
                                ;43142          ;1------------------------------; ODD ADDRESS
U 14AD, 0C80,0036,4030,0047,00FD,9 ;43143       NEXT/IE.CM.ODD                  ; BAD NEWS
                                ;43144
                                ;43145 CM.OS.WRT-ADEC.PC:
                                ;43146          ;------------------------------;
                                ;43147          VA_M[PC],                       ; PLACE ADDRESS IN VA
U 14DC, 0481,A002,403D,84A7,014A,6 ;43148       NEXT/CM.OS.WRT-ODD.CHECK1       ;
                                ;43149
                                ;43150 CM.OS.WRT-IMED:
                                ;43151          ;------------------------------;
                                ;43152          WB_SEXT(XB) PC_PC+2,            ; PUSH PC PAST IMMEDIATE DATA
U 14DD, 0881,7802,401D,A047,014A,6 ;43153       NEXT/CM.OS.WRT-ODD.CHECK1       ; WASTE A CYCLE
                                ;43154
                                ;43155 CM.OS.WRT-AUTO.DEC:
                                ;43156          ;------------------------------;
                                ;43157          R[GPR.R].SIZ_RB-D,              ; OPERAND ADDRESS IS GPR(RUNM) - SIZE
                                ;43158          SIZE[WORD],ODD ADDRESS?,        ; TEST FOR ODD ADDRESS
U 14DE, 0C82,0023,085C,C047,014A,C 347* ;43159  NEXT/CM.OS.WRT-ODD.CHECK4       ;
```

```
                                ;43160 .TOC "  Comp Mode Operand Specifier     : CM.OSR INDIRECT COMMON ROUTINE"
                                ;43161
                                ;43162 =0
                                ;43163 CM.OSR-GET.IND.1:
                                ;43164          ;0------------------------------;
                                ;43165          READ,                           ; GET ADDRESS OF ADDRESS OF OPERAND
                                ;43166          R[GPR.R].SIZ_RB+CONX(2),        ; ALWAYS AUTO-INC BY 2
                                ;43167          PUSH RBS+,                      ; SAVE REG INCASE OF FAILURE
                                ;43168          FLAG<1-0>?,                     ; TEST WHICH ROUTINE
U 14AE, 0483,573D,051C,C050,0148,8 ;43169       NEXT/CM.OSR.IND-TEST.FLAG
                                ;43170
                                ;43171          ;1------------------------------;
U 14AF, 0C80,0036,4030,0047,00FD,9 ;43172       NEXT/IE.CM.ODD                  ;
                                ;43173
                                ;43174 =00
                                ;43175 CM.OSR-GET.IND.2:
                                ;43176          ;00-----------------------------; NOT PC MODE
                                ;43177          VA<0>?,                         ; TEST FOR ODD ADDRESS
U 1478, 0081,B002,42BD,8047,014B,4 ;43178       NEXT/CM.OSR-GET.IND.TST         ;
                                ;43179
                                ;43180 =10      ;10-----------------------------; PC MODE
                                ;43181          Q_M[MDR],                       ; SAVE MDR
U 147A, 0C81,2592,5030,0047,014D,F ;43182       NEXT/CM.OSR-IND.DEC.PC          ;
                                ;43183
                                ;43184          ;11-----------------------------; NOT PC MODE
                                ;43185          VA<0>?,                         ; TEST FOR ODD ADDRESS
U 147B, 0081,B002,42BD,8047,014B,4 ;43186       NEXT/CM.OSR-GET.IND.TST         ;
                                ;43187
                                ;43188 CM.OSR-IND.DEC.PC:
                                ;43189          ;------------------------------;
                                ;43190          PC_M[PC]-ZLIT0[2],              ; AUTO-DEC THE PC
                                ;43191          FLAG<1-0>?,                     ; TEST WHICH ROUTINE
U 14DF, 0581,AC10,0530,1487,0148,8 ;43192       NEXT/CM.OSR.IND-TEST.FLAG
                                ;43193
                                ;43194 =00
                                ;43195 CM.OSR-GET.IND.3:
                                ;43196          ;00-----------------------------; NOT PC MODE
                                ;43197          VA_SEXT(XB)+R[GPR.R] PC_PC+2,   ; GET ADDRESS OF ADDRESS OF OPERAND
                                ;43198          WB<0>?,                         ; TEST FOR ODD ADDRESS
U 1484, 0481,7815,029C,E4A7,094B,4 349* ;43199  NEXT/CM.OSR-GET.IND.TST         ;
                                ;43200
                                ;43201 =10      ;10-----------------------------; PC MODE
                                ;43202          VA_PC+SEXT(XB)+2 PC_PC+2,       ; GET ADDRESS OF ADDRESS OF OPERAND
                                ;43203          WB<0>?,                         ; TEST FOR ODD ADDRESS
U 1486, 0881,7816,429D,A407,014B,4 ;43204       NEXT/CM.OSR-GET.IND.TST         ;
                                ;43205
                                ;43206          ;11-----------------------------; NOT PC MODE
                                ;43207          VA_SEXT(XB)+R[GPR.R] PC_PC+2,   ; GET ADDRESS OF ADDRESS OF OPERAND
                                ;43208          WB<0>?,                         ; TEST FOR ODD ADDRESS
U 1487, 0481,7815,029C,E4A7,094B,4 349* ;43209  NEXT/CM.OSR-GET.IND.TST         ;
```

```
                                ;43210 =0
                                ;43211 CM.OSR-GET.IND.TST:
                                ;43212          ;0-------------------------------;
                                ;43213          Q_M[MDR],                       ; SAVE MDR BEFORE DESTROYING
                                ;43214          READ,SIZE[IDEP],                ; GET ADDRESS OF OPERAND
                                ;43215          FLAG<1-0>?,                     ; TEST WHICH ROUTINE
U 14B4, 0481,2592,5530,0050,0148,8  ;43216      NEXT/CM.OSR.IND-TEST.FLAG
                                ;43217
                                ;43218          ;1-------------------------------;
U 14B5, 0C80,0036,4030,0047,00FD,9  ;43219      NEXT/IE.CM.ODD                  ;
                                ;43220
                                ;43221
                                ;43222 =00
                                ;43223 CM.OSR.IND-TEST.FLAG:
                                ;43224          ;00------------------------------; FROM OS.RED
                                ;43225          VA_ZEXT(M[MDR]),                ; PUT ADDRESS OF OPERAND INTO VA
                                ;43226          SIZE[WORD],ODD ADDRESS?,        ; TEST FOR ODD ADDRESS AND WROD INST
U 1488, 0C81,2016,485D,84A7,014B,C  ;43227      NEXT/CM.OSR-RED.END             ;
                                ;43228
                                ;43229          ;01------------------------------; FROM OS.MOD
                                ;43230          VA_ZEXT(M[MDR]),                ; PUT ADDRESS OF OPERAND INTO VA
                                ;43231          SIZE[WORD],ODD ADDRESS?,        ; TEST FOR ODD ADDRESS AND WROD INST
U 1489, 0481,2016,485D,84A7,014B,E  ;43232      NEXT/CM.OSR-MOD.END             ;
                                ;43233
                                ;43234          ;10------------------------------; FROM OS.WRT
                                ;43235          VA_MTEMP0_ZEXT(MDR),            ; PUT ADDRESS OF OPERAND INTO VA
                                ;43236          SIZE[WORD],ODD ADDRESS?,        ; TEST FOR ODD ADDRESS AND WROD INST
U 148A, 0487,2016,485D,84A7,014C,0  ;43237      NEXT/CM.OSR-WRT.END             ;
                                ;43238 =
                                ;43239 =0
                                ;43240 CM.OSR-RED.END:
                                ;43241          ;0-------------------------------;
                                ;43242          READ,SIZE[IDEP],                ; GET ACTUAL OPERAND
U 14BC, 0880,0036,4070,0050,0000,1  ;43243      IRDX [1]                        ;
                                ;43244
                                ;43245          ;1-------------------------------;
U 14BD, 0C80,0036,4030,0047,00FD,9  ;43246      NEXT/IE.CM.ODD                  ;
                                ;43247
                                ;43248 =0
                                ;43249 CM.OSR-MOD.END:
                                ;43250          ;0-------------------------------;
                                ;43251          READ.MOD,SIZE[IDEP],            ; GET ACTUAL OPERAND
U 14BE, 0080,0036,4070,0054,0000,1  ;43252      IRDX [1]                        ;
                                ;43253
                                ;43254          ;1-------------------------------;
U 14BF, 0C80,0036,4030,0047,00FD,9  ;43255      NEXT/IE.CM.ODD                  ;
                                ;43256
                                ;43257 =0
                                ;43258 CM.IRDX:
                                ;43259 CM.OSR-WRT.END:
                                ;43260          ;0-------------------------------;
U 14C0, 0880,0036,4070,0047,0000,1  ;43261      IRDX [1]                        ; NOTHING TO DO
                                ;43262
                                ;43263          ;1-------------------------------;
U 14C1, 0C80,0036,4030,0047,00FD,9  ;43264      NEXT/IE.CM.ODD                  ;
```

```
                                        ;43265 .TOC  ""      Compatibility Mode              : Ird Rom Definition"
;43266  .NOBIN
;43267
;43268  .CCODE
;43269
;43270  ;                       IRD1                    CNT0                    CNT1
;43271
;43272  0A3:    REG [CM.ADC-REG     ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;ADC
;43273          MEM [CM.OS.MOD      ] [CM.ADC-MEM     ] [IE.BAD.IRD     ]
;43274
;43275  0AB:    REG [CM.ADC-REG     ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;ADCB
;43276          MEM [CM.OS.MOD      ] [CM.ADC-MEM     ] [IE.BAD.IRD     ]
;43277
;43278  046:    REG [CM.OS.RED      ] [CM.ADD-REG     ] [IE.BAD.IRD     ],    ;ADD,1
;43279          MEM [CM.OS.RED      ] [CM.OS.MOD      ] [CM.ADD-MEM     ]
;43280
;43281  056:    REG [CM.OS.RED      ] [CM.ADD-REG     ] [IE.BAD.IRD     ],    ;ADD,2
;43282          MEM [CM.OS.RED      ] [CM.OS.MOD      ] [CM.ADD-MEM     ]
;43283
;43284  064:    REG [CM.OS.RED      ] [CM.ASH         ] [IE.BAD.IRD     ],    ;ASH,1
;43285          MEM [CM.OS.RED      ] [CM.ASH         ] [IE.BAD.IRD     ]
;43286
;43287  074:    REG [CM.OS.RED      ] [CM.ASH         ] [IE.BAD.IRD     ],    ;ASH,2
;43288          MEM [CM.OS.RED      ] [CM.ASH         ] [IE.BAD.IRD     ]
;43289
;43290  066:    REG [CM.OS.RED      ] [CM.ASHC        ] [IE.BAD.IRD     ],    ;ASHC,1
;43291          MEM [CM.OS.RED      ] [CM.ASHC        ] [IE.BAD.IRD     ]
;43292
;43293  076:    REG [CM.OS.RED      ] [CM.ASHC        ] [IE.BAD.IRD     ],    ;ASHC,2
;43294          MEM [CM.OS.RED      ] [CM.ASHC        ] [IE.BAD.IRD     ]
;43295
;43296  095:    REG [CM.ASL-REG     ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;ASL
;43297          MEM [CM.OS.MOD      ] [CM.ASL-MEM     ] [IE.BAD.IRD     ]
;43298
;43299  09D:    REG [CM.ASL-REG     ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;ASL
;43300          MEM [CM.OS.MOD      ] [CM.ASL-MEM     ] [IE.BAD.IRD     ]
;43301
;43302  094:    REG [CM.ASR-REG     ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;ASR
;43303          MEM [CM.OS.MOD      ] [CM.ASR-MEM     ] [IE.BAD.IRD     ]
;43304
;43305  09C:    REG [CM.ASR-REG     ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;ASRB
;43306          MEM [CM.OS.MOD      ] [CM.ASR-MEM     ] [IE.BAD.IRD     ]
;43307
;43308  01E:    REG [CM.BRCND       ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;BCC/BHIS
;43309          MEM [CM.BRCND       ] [IE.BAD.IRD     ] [IE.BAD.IRD     ]
;43310
;43311  01F:    REG [CM.BRCND       ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;BCS/BLO
;43312          MEM [CM.BRCND       ] [IE.BAD.IRD     ] [IE.BAD.IRD     ]
;43313
;43314  013:    REG [CM.BRCND       ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;BEQ
;43315          MEM [CM.BRCND       ] [IE.BAD.IRD     ] [IE.BAD.IRD     ]
;43316
;43317  014:    REG [CM.BRCND       ] [IE.BAD.IRD     ] [IE.BAD.IRD     ],    ;BGE
;43318          MEM [CM.BRCND       ] [IE.BAD.IRD     ] [IE.BAD.IRD     ]
```

```
;43319  ;                    IRD1                CNT0                CNT1
;43320
;43321  016:   REG [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;BGT
;43322         MEM [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
;43323
;43324  01A:   REG [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;BHI
;43325         MEM [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
;43326
;43327  044:   REG [CM.OS.RED         ] [CM.BIC-REG       ] [IE.BAD.IRD       ],    ;BIC,1
;43328         MEM [CM.OS.RED         ] [CM.OS.MOD        ] [CM.BIC-MEM       ]
;43329
;43330  054:   REG [CM.OS.RED         ] [CM.BIC-REG       ] [IE.BAD.IRD       ],    ;BIC,2
;43331         MEM [CM.OS.RED         ] [CM.OS.MOD        ] [CM.BIC-MEM       ]
;43332
;43333  04C:   REG [CM.OS.RED         ] [CM.BIC-REG       ] [IE.BAD.IRD       ],    ;BICB,1
;43334         MEM [CM.OS.RED         ] [CM.OS.MOD        ] [CM.BIC-MEM       ]
;43335
;43336  05C:   REG [CM.OS.RED         ] [CM.BIC-REG       ] [IE.BAD.IRD       ],    ;BICB,2
;43337         MEM [CM.OS.RED         ] [CM.OS.MOD        ] [CM.BIC-MEM       ]
;43338
;43339  045:   REG [CM.OS.RED         ] [CM.BIS-REG       ] [IE.BAD.IRD       ],    ;BIS,1
;43340         MEM [CM.OS.RED         ] [CM.OS.MOD        ] [CM.BIS-MEM       ]
;43341
;43342  055:   REG [CM.OS.RED         ] [CM.BIS-REG       ] [IE.BAD.IRD       ],    ;BIS,2
;43343         MEM [CM.OS.RED         ] [CM.OS.MOD        ] [CM.BIS-MEM       ]
;43344
;43345  04D:   REG [CM.OS.RED         ] [CM.BIS-REG       ] [IE.BAD.IRD       ],    ;BISB,1
;43346         MEM [CM.OS.RED         ] [CM.OS.MOD        ] [CM.BIS-MEM       ]
;43347
;43348  05D:   REG [CM.OS.RED         ] [CM.BIS-REG       ] [IE.BAD.IRD       ],    ;BISB,2
;43349         MEM [CM.OS.RED         ] [CM.OS.MOD        ] [CM.BIS-MEM       ]
;43350
;43351  043:   REG [CM.OS.RED         ] [CM.BIT-REG       ] [IE.BAD.IRD       ],    ;BIT,1
;43352         MEM [CM.OS.RED         ] [CM.OS.RED        ] [CM.BIT-MEM       ]
;43353
;43354  053:   REG [CM.OS.RED         ] [CM.BIT-REG       ] [IE.BAD.IRD       ],    ;BIT,2
;43355         MEM [CM.OS.RED         ] [CM.OS.RED        ] [CM.BIT-MEM       ]
;43356
;43357  04B:   REG [CM.OS.RED         ] [CM.BIT-REG       ] [IE.BAD.IRD       ],    ;BITB,1
;43358         MEM [CM.OS.RED         ] [CM.OS.RED        ] [CM.BIT-MEM       ]
;43359
;43360  05B:   REG [CM.OS.RED         ] [CM.BIT-REG       ] [IE.BAD.IRD       ],    ;BITB,2
;43361         MEM [CM.OS.RED         ] [CM.OS.RED        ] [CM.BIT-MEM       ]
;43362
;43363  017:   REG [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;BLE
;43364         MEM [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
;43365
;43366  01B:   REG [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;BLOS
;43367         MEM [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
;43368
;43369  015:   REG [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;BLT
;43370         MEM [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
;43371
;43372  019:   REG [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;BMI
;43373         MEM [CM.BRCND          ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
```

```
;43374  ;                    IRD1                    CNT0                    CNT1
;43375
;43376  012:    REG [CM.BRCND          ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;BNE
;43377          MEM [CM.BRCND          ] [IE.BAD.IRD          ] [IE.BAD.IRD          ]
;43378
;43379  018:    REG [CM.BRCND          ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;BPL
;43380          MEM [CM.BRCND          ] [IE.BAD.IRD          ] [IE.BAD.IRD          ]
;43381
;43382  0C3:    REG [IE.CM.BPT         ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;BPT *
;43383          MEM [IE.CM.BPT         ] [IE.BAD.IRD          ] [IE.BAD.IRD          ]
;43384
;43385  011:    REG [CM.BR             ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;BR
;43386          MEM [CM.BR             ] [IE.BAD.IRD          ] [IE.BAD.IRD          ]
;43387
;43388  01C:    REG [CM.BRCND          ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;BVC
;43389          MEM [CM.BRCND          ] [IE.BAD.IRD          ] [IE.BAD.IRD          ]
;43390
;43391  01D:    REG [CM.BRCND          ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;BVS
;43392          MEM [CM.BRCND          ] [IE.BAD.IRD          ] [IE.BAD.IRD          ]
;43393
;43394  0F2:    REG [CM-SET.CLEAR      ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;CC-CLR
;43395          MEM [CM-SET.CLEAR      ] [IE.BAD.IRD          ] [IE.BAD.IRD          ]
;43396
;43397  0F6:    REG [CM-SET.CLEAR      ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;CC-SET
;43398          MEM [CM-SET.CLEAR      ] [IE.BAD.IRD          ] [IE.BAD.IRD          ]
;43399
;43400  082:    REG [CM.CLR-REG        ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;CLR
;43401          MEM [CM.OS.WRT         ] [CM.CLR-MEM          ] [IE.BAD.IRD          ]
;43402
;43403  08A:    REG [CM.CLR-REG        ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;CLRB
;43404          MEM [CM.OS.WRT         ] [CM.CLR-MEM          ] [IE.BAD.IRD          ]
;43405
;43406  042:    REG [CM.OS.RED         ] [CM.CMP-REG          ] [IE.BAD.IRD          ],    ;CMP,1
;43407          MEM [CM.OS.RED         ] [CM.OS.RED           ] [CM.CMP-MEM          ]
;43408
;43409  052:    REG [CM.OS.RED         ] [CM.CMP-REG          ] [IE.BAD.IRD          ],    ;CMP,2
;43410          MEM [CM.OS.RED         ] [CM.OS.RED           ] [CM.CMP-MEM          ]
;43411
;43412  04A:    REG [CM.OS.RED         ] [CM.CMP-REG          ] [IE.BAD.IRD          ],    ;CMPB,1
;43413          MEM [CM.OS.RED         ] [CM.OS.RED           ] [CM.CMP-MEM          ]
;43414
;43415  05A:    REG [CM.OS.RED         ] [CM.CMP-REG          ] [IE.BAD.IRD          ],    ;CMPB,2
;43416          MEM [CM.OS.RED         ] [CM.OS.RED           ] [CM.CMP-MEM          ]
;43417
;43418  083:    REG [CM.COM-REG        ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;COM
;43419          MEM [CM.OS.MOD         ] [CM.COM-MEM          ] [IE.BAD.IRD          ]
;43420
;43421  08B:    REG [CM.COM-REG        ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;COMB
;43422          MEM [CM.OS.MOD         ] [CM.COM-MEM          ] [IE.BAD.IRD          ]
;43423
;43424  093:    REG [CM.DEC-REG        ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;DEC
;43425          MEM [CM.OS.MOD         ] [CM.DEC-MEM          ] [IE.BAD.IRD          ]
;43426
;43427  09B:    REG [CM.DEC-REG        ] [IE.BAD.IRD          ] [IE.BAD.IRD          ],    ;DECB
;43428          MEM [CM.OS.MOD         ] [CM.DEC-MEM          ] [IE.BAD.IRD          ]
```

```
;43429  ;                    IRD1                    CNT0                    CNT1
;43430
;43431  062:   REG [CM.OS.RED      ] [CM.DIV        ] [IE.BAD.IRD    ],   ;DIV,1
;43432         MEM [CM.OS.RED      ] [CM.DIV        ] [IE.BAD.IRD    ]
;43433
;43434  072:   REG [CM.OS.RED      ] [CM.DIV        ] [IE.BAD.IRD    ],   ;DIV,2
;43435         MEM [CM.OS.RED      ] [CM.DIV        ] [IE.BAD.IRD    ]
;43436
;43437  088:   REG [IE.CM.EMT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;EMT,1
;43438         MEM [IE.CM.EMT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ]
;43439
;43440  089:   REG [IE.CM.EMT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;EMT,2
;43441         MEM [IE.CM.EMT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ]
;43442
;43443  098:   REG [IE.CM.EMT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;EMT,3
;43444         MEM [IE.CM.EMT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ]
;43445
;43446  099:   REG [IE.CM.EMT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;EMT,4
;43447         MEM [IE.CM.EMT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ]
;43448
;43449  092:   REG [CM.INC-REG     ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;INC
;43450         MEM [CM.OS.MOD      ] [CM.INC-MEM    ] [IE.BAD.IRD    ]
;43451
;43452  09A:   REG [CM.INC-REG     ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;INCB
;43453         MEM [CM.OS.MOD      ] [CM.INC-MEM    ] [IE.BAD.IRD    ]
;43454
;43455  0C4:   REG [IE.CM.IOT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;IOT *
;43456         MEM [IE.CM.IOT      ] [IE.BAD.IRD    ] [IE.BAD.IRD    ]
;43457
;43458  0E1:   REG [CM.JMP-REG     ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;JMP,1
;43459         MEM [CM.OS.WRT      ] [CM.JMP-MEM    ] [IE.BAD.IRD    ]
;43460
;43461  0E3:   REG [CM.JMP-REG     ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;JMP,2
;43462         MEM [CM.OS.WRT      ] [CM.JMP-MEM    ] [IE.BAD.IRD    ]
;43463
;43464  0E5:   REG [CM.JMP-REG     ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;JMP,3
;43465         MEM [CM.OS.WRT      ] [CM.JMP-MEM    ] [IE.BAD.IRD    ]
;43466
;43467  0E7:   REG [CM.JMP-REG     ] [IE.BAD.IRD    ] [IE.BAD.IRD    ],   ;JMP,4
;43468         MEM [CM.OS.WRT      ] [CM.JMP-MEM    ] [IE.BAD.IRD    ]
;43469
;43470  080:   REG [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ],   ;JSR,1
;43471         MEM [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ]
;43472
;43473  081:   REG [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ],   ;JSR,2
;43474         MEM [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ]
;43475
;43476  090:   REG [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ],   ;JSR,3
;43477         MEM [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ]
;43478
;43479  091:   REG [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ],   ;JSR,4
;43480         MEM [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ]
;43481
;43482  0A0:   REG [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ],   ;JSR,5
;43483         MEM [CM.OS.WRT      ] [CM.JSR        ] [IE.BAD.IRD    ]
```

```
;43484  ;                    IRD1                    CNT0                    CNT1
;43485
;43486  0A1:    REG [CM.OS.WRT          ] [CM.JSR           ] [IE.BAD.IRD       ],    ;JSR,6
;43487          MEM [CM.OS.WRT          ] [CM.JSR           ] [IE.BAD.IRD       ]
;43488
;43489  0B0:    REG [CM.OS.WRT          ] [CM.JSR           ] [IE.BAD.IRD       ],    ;JSR,7
;43490          MEM [CM.OS.WRT          ] [CM.JSR           ] [IE.BAD.IRD       ]
;43491
;43492  0B1:    REG [CM.OS.WRT          ] [CM.JSR           ] [IE.BAD.IRD       ],    ;JSR,.
;43493          MEM [CM.OS.WRT          ] [CM.JSR           ] [IE.BAD.IRD       ]
;43494
;43495  0AD:    REG [CM.MFPD-REG        ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;MFPD
;43496          MEM [CM.OS.RED          ] [CM.MFPD-MEM      ] [IE.BAD.IRD       ]
;43497
;43498  0A5:    REG [CM.MFPI-REG        ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;MFPI
;43499          MEM [CM.OS.RED          ] [CM.MFPI-MEM      ] [IE.BAD.IRD       ]
;43500
;43501  0BC:    REG [CM.MTPD            ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;MTPD
;43502          MEM [CM.OS.WRT          ] [CM.MTPD          ] [IE.BAD.IRD       ]
;43503
;43504  0B4:    REG [CM.MTPI            ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;MTPI
;43505          MEM [CM.OS.WRT          ] [CM.MTPI          ] [IE.BAD.IRD       ]
;43506
;43507  041:    REG [CM.OS.RED          ] [CM.MOV-REG       ] [IE.BAD.IRD       ],    ;MOV,1
;43508          MEM [CM.OS.RED          ] [CM.OS.WRT        ] [CM.MOV-MEM       ]
;43509
;43510  051:    REG [CM.OS.RED          ] [CM.MOV-REG       ] [IE.BAD.IRD       ],    ;MOV,2
;43511          MEM [CM.OS.RED          ] [CM.OS.WRT        ] [CM.MOV-MEM       ]
;43512
;43513  049:    REG [CM.OS.RED          ] [CM.MOVB-REG      ] [IE.BAD.IRD       ],    ;MOVB,1
;43514          MEM [CM.OS.RED          ] [CM.OS.WRT        ] [CM.MOV-MEM       ]
;43515
;43516  059:    REG [CM.OS.RED          ] [CM.MOVB-REG      ] [IE.BAD.IRD       ],    ;MOVB,2
;43517          MEM [CM.OS.RED          ] [CM.OS.WRT        ] [CM.MOV-MEM       ]
;43518
;43519  060:    REG [CM.OS.RED          ] [CM.MUL           ] [IE.BAD.IRD       ],    ;MUL,1
;43520          MEM [CM.OS.RED          ] [CM.MUL           ] [IE.BAD.IRD       ]
;43521
;43522  070:    REG [CM.OS.RED          ] [CM.MUL           ] [IE.BAD.IRD       ],    ;MUL,2
;43523          MEM [CM.OS.RED          ] [CM.MUL           ] [IE.BAD.IRD       ]
;43524
;43525  042:    REG [CM.NEG-REG         ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;NEG
;43526          MEM [CM.OS.MOD          ] [CM.NEG-MEM       ] [IE.BAD.IRD       ]
;43527
;43528  0AA:    REG [CM.NEG-REG         ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;NEGB
;43529          MEM [CM.OS.MOD          ] [CM.NEG-MEM       ] [IE.BAD.IRD       ]
;43530
;43531  000:    REG [IE.CM.RESOP        ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;RESVRD
;43532          MEM [IE.CM.RESOP        ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
;43533
;43534  001:    REG [IE.CM.RESOP        ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;RESVRD
;43535          MEM [IE.CM.RESOP        ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
;43536
;43537  002:    REG [IE.CM.RESOP        ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;RESVRD
;43538          MEM [IE.CM.RESOP        ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
```

```
;43539  ;                    IRD1                    CNT0                    CNT1
;43540
;43541  003:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43542          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43543
;43544  004:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43545          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43546
;43547  005:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43548          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43549
;43550  006:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43551          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43552
;43553  007:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43554          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43555
;43556  008:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43557          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43558
;43559  009:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43560          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43561
;43562  00A:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43563          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43564
;43565  00B:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43566          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43567
;43568  00C:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43569          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43570
;43571  00D:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43572          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43573
;43574  00E:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43575          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43576
;43577  00F:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43578          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43579
;43580  010:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43581          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43582
;43583  020:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43584          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43585
;43586  021:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43587          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43588
;43589  022:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43590          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43591
;43592  023:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43593          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
```

```
;43594  ;                    IRD1                    CNT0                    CNT1
;43595
;43596  024:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43597          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43598
;43599  025:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43600          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43601
;43602  026:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43603          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43604
;43605  027:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43606          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43607
;43608  028:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43609          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43610
;43611  029:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43612          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43613
;43614  02A:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43615          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43616
;43617  02B:    REG [IE.CM.RESOP        [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43618          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43619
;43620  02C:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43621          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43622
;43623  02D:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43624          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43625
;43626  02E:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43627          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43628
;43629  02F:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43630          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43631
;43632  030:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43633          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43634
;43635  031:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43636          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43637
;43638  032:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43639          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43640
;43641  033:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43642          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43643
;43644  034:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43645          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43646
;43647  035:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43648          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
```

```
;43649  ;                       IRD1                    CNT0                    CNT1
;43650
;43651  036:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43652          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43653
;43654  037:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43655          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43656
;43657  038:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43658          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43659
;43660  039:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43661          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43662
;43663  03A:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43664          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43665
;43666  03B:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43667          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43668
;43669  03C:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43670          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43671
;43672  03D:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43673          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43674
;43675  03F:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43676          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43677
;43678  03F:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43679          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43680
;43681  040:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43682          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43683
;43684  047:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43685          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43686
;43687  048:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43688          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43689
;43690  04F:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43691          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43692
;43693  050:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43694          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43695
;43696  057:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43697          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43698
;43699  058:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43700          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43701
;43702  05F:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43703          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
```

```
;43704  ;                    IRD1                    CNT0                    CNT1
;43705
;43706  063:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43707          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43708
;43709  065:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43710          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43711
;43712  068:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43713          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43714
;43715  069:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43716          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43717
;43718  06A:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43719          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43720
;43721  06B:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43722          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43723
;43724  06C:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43725          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43726
;43727  06D:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43728          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43729
;43730  06E:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43731          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43732
;43733  06F:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43734          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43735
;43736  073:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43737          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43738
;43739  075:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43740          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43741
;43742  078:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43743          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43744
;43745  079:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43746          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43747
;43748  07A:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43749          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43750
;43751  07B:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43752          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43753
;43754  07C:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43755          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43756
;43757  07D:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43758          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
```

```
;43759  ;                    IRD1                    CNT0                    CNT1
;43760
;43761  07E:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43762          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43763
;43764  07F:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43765          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43766
;43767  086:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43768          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43769
;43770  087:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43771          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43772
;43773  08E:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43774          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43775
;43776  08F:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43777          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43778
;43779  096:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43780          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43781
;43782  097:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43783          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43784
;43785  09E:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43786          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43787
;43788  09F:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43789          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43790
;43791  0A4:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43792          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43793
;43794  0A6:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43795          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43796
;43797  0A7:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43798          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43799
;43800  0AC:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43801          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43802
;43803  0AE:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43804          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43805
;43806  0AF:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43807          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43808
;43809  0B6:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43810          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43811
;43812  0B7:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],     ;RESVRD
;43813          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
```

```
;43814  ;                    IRD1                    CNT0                    CNT1
;43815
;43816  0BD:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43817          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43818
;43819  0BE:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43820          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43821
;43822  0BF:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43823          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43824
;43825  0C0:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43826          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43827
;43828  0C1:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43829          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43830
;43831  0C5:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43832          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43833
;43834  0C7:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43835          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43836
;43837  0C8:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43838          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43839
;43840  0C9:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43841          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43842
;43843  0CA:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43844          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43845
;43846  0CB:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43847          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43848
;43849  0CC:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43850          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43851
;43852  0CD:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43853          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43854
;43855  0CE:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43856          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43857
;43858  0CF:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43859          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43860
;43861  0D0:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43862          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43863
;43864  0D1:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43865          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
;43866
;43867  0D2:    REG [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ],     ;RESVRD
;43868          MEM [IE.CM.RESOP        ] [IE.BAD.IRD        ] [IE.BAD.IRD        ]
```

```
;43869  ;                    IRD1                    CNT0                    CNT1
;43870
;43871  0D3:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43872          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43873
;43874  0D4:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43875          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43876
;43877  0D5:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43878          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43879
;43880  0D6:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43881          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43882
;43883  0D7:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43884          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43885
;43886  0D8:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43887          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43888
;43889  0D9:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43890          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43891
;43892  0DA:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43893          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43894
;43895  0DB:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43896          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43897
;43898  0DC:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43899          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43900
;43901  0DD:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43902          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43903
;43904  0DE:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43905          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43906
;43907  0DF:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43908          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43909
;43910  0E0:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43911          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43912
;43913  0E2:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43914          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43915
;43916  0E4:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43917          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43918
;43919  0E6:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43920          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
;43921
;43922  0E8:    REG [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ],    ;RESVRD
;43923          MEM [IE.CM.RESOP      ] [IE.BAD.IRD      ] [IE.BAD.IRD      ]
```

```
;43924  ;                       IRD1                    CNT0                    CNT1
;43925
;43926  0E9:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43927          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43928
;43929  0EA:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43930          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43931
;43932  0EB:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43933          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
 43934
;43935  0EC:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43936          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43937
;43938  0ED:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43939          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
 43940
;43941  0EE:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43942          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43943
;43944  0EF:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43945          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43946
;43947  0F4:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43948          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43949
;43950  0F8:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43951          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43952
;43953  0F9:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43954          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43955
;43956  0FA:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43957          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43958
;43959  0FB:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43960          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43961
;43962  0FC:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43963          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43964
;43965  0FD:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43966          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43967
;43968  0FE:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43969          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43970
;43971  0FF:    REG [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RESVRD
;43972          MEM [IE.CM.RESOP        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43973
;43974  085:    REG [CM.ROL-REG         ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;ROL
;43975          MEM [CM.OS.MOD          ] [CM.ROL-MEM           ] [IE.BAD.IRD           ]
;43976
;43977  08D:    REG [CM.ROL-REG         ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;ROLB
;43978          MEM [CM.OS.MOD          ] [CM.ROL-MEM           ] [IE.BAD.IRD           ]
```

```
;43979  ;                       IRD1                    CNT0                    CNT1
;43980
;43981  084:    REG [CM.ROR-REG         ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;ROR
;43982          MEM [CM.OS.MOD          ] [CM.ROR-MEM           ] [IE.BAD.IRD           ]
;43983
;43984  08C:    REG [CM.RORB-REG        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RORB
;43985          MEM [CM.OS.MOD          ] [CM.RORB-MEM          ] [IE.BAD.IRD           ]
;43986
;43987  0C2:    REG [CM.RTI             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RTI *
;43988          MEM [CM.RTI             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43989
;43990  0F0:    REG [CM.RTS             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RTS *
;43991          MEM [CM.RTS             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43992
;43993  0C6:    REG [CM.RTT             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;RTT *
;43994          MEM [CM.RTT             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;43995
;43996  0B2:    REG [CM.SBC-REG         ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;SBC
;43997          MEM [CM.OS.MOD          ] [CM.SBC-MEM           ] [IE.BAD.IRD           ]
;43998
;43999  0BA:    REG [CM.SBC-REG         ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;SBCB
;44000          MEM [CM.OS.MOD          ] [CM.SBC-MEM           ] [IE.BAD.IRD           ]
;44001
;44002  067:    REG [CM.SOB             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;SOB,1
;44003          MEM [CM.SOB             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;44004
;44005  077:    REG [CM.SOB             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;SOB,2
;44006          MEM [CM.SOB             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;44007
;44008  04E:    REG [CM.OS.RED          ] [CM.SUB-REG           ] [IE.BAD.IRD           ],      ;SUB,1
;44009          MEM [CM.OS.RED          ] [CM.OS.MOD            ] [CM.SUB-MEM           ]
;44010
;44011  05E:    REG [CM.OS.RED          ] [CM.SUB-REG           ] [IE.BAD.IRD           ],      ;SUB,2
;44012          MEM [CM.OS.RED          ] [CM.OS.MOD            ] [CM.SUB-MEM           ]
;44013
;44014  0F1:    REG [CM.SWAB-REG        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;SWAB,1
;44015          MEM [CM.OS.MOD          ] [CM.SWAB-MEM          ] [IE.BAD.IRD           ]
;44016
;44017  0F3:    REG [CM.SWAB-REG        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;SWAB,1
;44018          MEM [CM.OS.MOD          ] [CM.SWAB-MEM          ] [IE.BAD.IRD           ]
;44019
;44020  0F5:    REG [CM.SWAB-REG        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;SWAB,1
;44021          MEM [CM.OS.MOD          ] [CM.SWAB-MEM          ] [IE.BAD.IRD           ]
;44022
;44023  0F7:    REG [CM.SWAB-REG        ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;SWAB,1
;44024          MEM [CM.OS.MOD          ] [CM.SWAB-MEM          ] [IE.BAD.IRD           ]
;44025
;44026  0B5:    REG [CM.SXT             ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;SXT
;44027          MEM [CM.OS.WRT          ] [CM.SXT               ] [IE.BAD.IRD           ]
;44028
;44029  0A8:    REG [IE.CM.TRAP         ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;TRAP,1
;44030          MEM [IE.CM.TRAP         ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
;44031
;44032  0A9:    REG [IE.CM.TRAP         ] [IE.BAD.IRD           ] [IE.BAD.IRD           ],      ;TRAP,2
;44033          MEM [IE.CM.TRAP         ] [IE.BAD.IRD           ] [IE.BAD.IRD           ]
```

```
;44034  ;                    IRD1                    CNT0                    CNT1
;44035
;44036  0B8:    REG [IE.CM.TRAP       ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;TRAP,3
;44037          MEM [IE.CM.TRAP       ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
;44038
;44039  0B9:    REG [IE.CM.TRAP       ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;TRAP,4
;44040          MEM [IE.CM.TRAP       ] [IE.BAD.IRD       ] [IE.BAD.IRD       ]
;44041
;44042  0B3:    REG [CM.TST-REG       ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;TST
;44043          MEM [CM.OS.RED        ] [CM.TST-MEM       ] [IE.BAD.IRD       ]
;44044
;44045  0BB:    REG [CM.TST-REG       ] [IE.BAD.IRD       ] [IE.BAD.IRD       ],    ;TSTB
;44046          MEM [CM.OS.RED        ] [CM.TST-MEM       ] [IE.BAD.IRD       ]
;44047
;44048  061:    REG [CM.XOR           ] [CM.OS.MOD        ] [CM.XOR-WRITE     ],    ;XOR,1
;44049          MEM [CM.XOR           ] [CM.OS.MOD        ] [CM.XOR-WRITE     ]
;44050
;44051  071:    REG [CM.XOR           ] [CM.OS.MOD        ] [CM.XOR-WRITE     ],    ;XOR,2
;44052          MEM [CM.XOR           ] [CM.OS.MOD        ] [CM.XOR-WRITE     ]
```

```
;44053          .TOC    "      Compatibility Mode              : Dsize Rom Definition"
;44054          .DCODE
;44055  1A3:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ADC
;44056  1AB:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ADCB
;44057  146:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ADD,1
;44058  156:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ADD,2
;44059  164:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ASH,1
;44060  174:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ASH,2
;44061  165:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ASHC,1
;44062  175:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ASHC,2
;44063  195:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ASL
;44064  19D:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ASLB
;44065  194:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ASR
;44066  19C:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ASRB
;44067  11E:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BCC/BHIS
;44068  11F:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BCS/BLO
;44069  113:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BEQ
;44070  114:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BGE
;44071  116:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BGT
;44072  11A:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BHI
;44073  144:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BIC,1
;44074  154:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BIC,2
;44075  14C:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BICB,1
;44076  15C:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BICB,2
;44077  145:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BIS,1
;44078  155:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BIS,2
;44079  14D:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BISB,1
;44080  15D:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BISB,2
;44081  143:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BIT,1
;44082  153:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BIT,2
;44083  14B:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BITB,1
;44084  15B:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BITB,2
;44085  117:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BLE
;44086  11B:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BLOS
;44087  115:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BLT
;44088  119:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BMI
;44089  112:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BNE
;44090  118:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BPL
;44091  1C3:    SIZE [0     ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BPT *
;44092  111:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BR
;44093  11C:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BVC
;44094  11D:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;BVS
;44095  1F2:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;CC-CLR
;44096  1F6:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;CC-SET
;44097  182:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;CLR
;44098  18A:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;CLRB
;44099  142:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;CMP,1
;44100  152:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;CMP,2
;44101  14A:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;CMPB,1
;44102  15A:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;CMPB,2
;44103  183:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;COM
;44104  18B:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;COMB
;44105  193:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;DEC
;44106  19B:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;DECB
;44107  162:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;DIV,1
```

```
;44108  172:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;DIV,2
;44109  188:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;EMT,1
;44110  189:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;EMT,2
;44111  198:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;EMT,3
;44112  199:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;EMT,4
;44113  192:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;INC
;44114  19A:    SIZE [BYTE] [0    ] [0    ] [0    ] [0    ] [0    ]  ;INCB
;44115  1C4:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;IOT *
;44116  1E1:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JMP,1
;44117  1E3:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JMP,2
;44118  1E5:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JMP,3
;44119  1E7:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JMP,4
;44120  180:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JSR,1
;44121  181:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JSR,2
;44122  190:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JSR,3
;44123  191:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JSR,4
;44124  1A0:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JSR,5
;44125  1A1:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JSR,6
;44126  1B0:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JSR,7
;44127  1B1:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;JSR,8
;44128  1AD:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MFPD
;44129  1A5:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MFPI
;44130  1BC:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MTPD
;44131  1B4:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MTPI
;44132  141:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MOV,1
;44133  151:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MOV,2
;44134  149:    SIZE [BYTE] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MOVB,1
;44135  159:    SIZE [BYTE] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MOVB,2
;44136  160:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MUL,1
;44137  170:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;MUL,2
;44138  1A2:    SIZE [WORD] [0    ] [0    ] [0    ] [0    ] [0    ]  ;NEG
;44139  1AA:    SIZE [BYTE] [0    ] [0    ] [0    ] [0    ] [0    ]  ;NEGB
;44140  100:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44141  101:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44142  102:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44143  103:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44144  104:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44145  105:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44146  106:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44147  107:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44148  108:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44149  109:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44150  10A:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44151  10B:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44152  10C:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44153  10D:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44154  10E:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44155  10F:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44156  110:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44157  120:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44158  121:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44159  122:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44160  123:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44161  124:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
;44162  125:    SIZE [0    ] [0    ] [0    ] [0    ] [0    ] [0    ]  ;RESVRD
```

```
;44163  126:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44164  127:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44165  128:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44166  129:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44167  12A:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44168  12B:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44169  12C:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44170  12D:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44171  12E:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44172  12F:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44173  130:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44174  131:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44175  132:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44176  133:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44177  134:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44178  135:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44179  136:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44180  137:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44181  138:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44182  139:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44183  13A:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44184  13B:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44185  13C:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44186  13D:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44187  13E:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44188  13F:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44189  140:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44190  147:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44191  148:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44192  14F:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44193  150:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44194  157:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44195  158:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44196  15F:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44197  163:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44198  165:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44199  168:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44200  169:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44201  16A:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44202  16B:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44203  16C:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44204  16D:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44205  16E:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44206  16F:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44207  173:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44208  175:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44209  178:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44210  179:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44211  17A:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44212  17B:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44213  17C:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44214  17D:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44215  17E:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44216  17F:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
;44217  186:    SIZE [0  ] [0  ] [0  ] [0  ] [0  ] [0  ] [0  ]  ;RESVRD
```

```
;44218  187:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44219  18E:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44220  18F:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44221  196:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44222  197:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44223  19E:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44224  19F:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44225  1A4:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44226  1A6:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44227  1A7:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44228  1AC:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44229  1AE:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44230  1AF:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44231  1B6:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44232  1B7:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44233  1BD:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44234  1BE:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44235  1BF:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44236  1C0:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44237  1C1:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44238  1C5:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44239  1C7:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44240  1C8:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44241  1C9:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44242  1CA:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44243  1CB:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44244  1CC:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44245  1CD:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44246  1CE:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44247  1CF:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44248  1D0:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44249  1D1:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44250  1D2:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44251  1D3:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44252  1D4:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44253  1D5:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44254  1D6:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44255  1D7:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44256  1D8:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44257  1D9:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44258  1DA:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44259  1DB:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44260  1DC:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44261  1DD:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44262  1DE:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44263  1DF:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44264  1E0:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44265  1E2:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44266  1E4:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44267  1E6:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44268  1E8:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44269  1E9:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44270  1EA:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44271  1EB:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44272  1EC:     SIZE [0   ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
```

```
;44273  1ED:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44274  1EE:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44275  1EF:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44276  1F4:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44277  1F8:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44278  1F9:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44279  1FA:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44280  1FB:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44281  1FC:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44282  1FD:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44283  1FE:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44284  1FF:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RESVRD
;44285  185:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ROL
;44286  18D:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ROLB
;44287  184:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;ROR
;44288  18C:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RORB
;44289  1C2:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RTI *
;44290  1F0:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RTS *
;44291  1C6:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;RTT *
;44292  1B2:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SBC
;44293  1BA:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SBCB
;44294  167:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SOB,1
;44295  177:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SOB,2
;44296  14E:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SUB,1
;44297  15E:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SUB,2
;44298  1F1:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SWAB,1
;44299  1F3:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SWAB,2
;44300  1F5:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SWAB,3
;44301  1F7:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SWAB,4
;44302  1B5:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;SXT
;44303  1A8:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;TRAP,1
;44304  1A9:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;TRAP,2
;44305  1B8:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;TRAP,3
;44306  1B9:    SIZE [0    ] [0   ] [0   ] [0   ] [0   ] [0   ]  ;TRAP,4
;44307  1B3:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;TST
;44308  1BB:    SIZE [BYTE] [0   ] [0   ] [0   ] [0   ] [0   ]  ;TSTB
;44309  161:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;XOR,1
;44310  171:    SIZE [WORD] [0   ] [0   ] [0   ] [0   ] [0   ]  ;XOR,2
;44311  .UCODE
                                        ;44312 .BIN
```

```
                                        ;44313 .TOC "OSR.MIC"
                                        ;44314 .TOC "Revision 31.0"
                                        ;44315 ;       Jeff Peng, Gerard Koeckhoven, Brian Allison
                                        ;44316
;44317  .NOBIN
;44318  .TOC "  Revision History"
;44319
;44320  ; REV   EXPLAINATION
;44321
;44322  ; 31    Fix OS.ADD_INC.PC for instruction PUSHAQ & MOVAQ
;44323  ; 30    Code around an undocumented hardware conflict that causes
;44324  ;       MDR_0 and IRDX in the same cycle not to work properly,
;44325  ;       causing QUAD and DOUBLE short literals not to work properly
;44326  ;       when they are the last byte on a page and there
;44327  ;       is a IB MISS on the next page.
;44328  ;29    29-MAY-1980
;44329  ;       Fix indexed immediate in OS.BOA.QUAD.
;44330  ;28    Fix indexed immediate.
;44331  ;       Change immediate mode for write type operands to take an addressing
;44332  ;       mode fault.
;44333  ;27    Initial release.
                                        ;44334 .BIN
```

```
                                ;44335 .TOC "  Native Mode Operand Specifier   : OS.RED"
                                ;44336
                                ;44337 ;*****************************************************************************
                                ;44338 ;       MDR is saved in the Q-register and is always destroyed.
                                ;44339 ;       Evaluates read type operands and places the result in MDR.
                                ;44340 ;       MTEMP0 is destroyed.
                                ;44341 ;       Read access for the operand is checked.
                                ;44342 ;*****************************************************************************
                                ;44343
                                ;44344 .REGION/OSR.R1L,OSR.R1H/OSR.R2L,OSR.R2H/OSR.R3L,OSR.R3H
                                ;44345 100:
                                ;44346 OS.RED:
                                ;44347         ;0000-------------------------; Rn        REGISTER MODE
                                ;44348         FPA_Q_M[MDR] MDR_R[GPR.R],     ; PLACE OPERAND (GPR(RNUM)) IN MDR
U 0100, 0487,2004,707C,C467,1000,1              ;44349         CLOBBER MTEMP0 DEF,IRDX [1]    ; SAVE MDR IN Q BEFORE CLOBBERING IT
                                ;44350
                                ;44351 101:     ;0001-------------------------; (Rn)+     AUTOINCREMENT MODE
                                ;44352         FPA_Q_M[MDR] VA_R[GPR.R],      ; PLACE ADDRESS OF OPERAND IN VA
                                ;44353         CLOBBER MTEMP0 DEF,            ; PUT GARBAGE IN MTEMP0
U 0101, 0887,2004,703C,C4A7,1048,C              ;44354         NEXT/OS.RED-AUTO.INC           ; INC GPR(RNUM) BY SIZE
                                ;44355
                                ;44356 102:     ;0010-------------------------; I^#cons   IMMEDIATE MODE
                                ;44357         Q_XB_PC_PC+I,                  ; GET PROPER AMOUNT OF IMMEDIATE DATA
U 0102, 0481,7002,503D,A047,0048,E              ;44358         NEXT/OS.RED-IMMED              ;
                                ;44359
                                ;44360 103:     ;0011-------------------------; S^#cons   LITERAL MODE
                                ;44361         FPA_Q_M[MDR].LITNXT,           ; PASS OPERAND TO FPA
                                ;44362         MDR_ZEXT(OSR),                 ; MDR GETS OPERAND FROM I-STREAM
                                ;44363         CLOBBER MTEMP0 DEF,            ; PUT GARBAGE IN MTEMP0
U 0103, 0087,2592,5070,05E7,2000,1              ;44364         IRDX [1]                       ; SAVE MDR IN Q
                                ;44365
                                ;44366 104:     ;0100-------------------------; -(Rn)     AUTODECREMENT MODE
                                ;44367         VA_R[GPR.R]_RB-CONX.SIZ,       ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
                                ;44368         PUSH RBS-,                     ; VA GETS OPERAND ADDRESS
U 0104, 0085,473C,003C,C4A7,0049,4              ;44369         NEXT/OS.RED-READ.SAV.EXIT      ; RBS GETS !RNUM!SIZE!-!
                                ;44370
                                ;44371 105:     ;0101-------------------------; Addr      PELATIVE MODE
                                ;44372         VA_PC+SEXT(XB)+I_PC_PC+I,      ; ADDRESS OF OPERAND IS PC + DSPLMT
                                ;44373         SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 0105, 0487,7816,403D,8407,0049,4              ;44374         NEXT/OS.RED-READ.SAV.EXIT      ;
                                ;44375
                                ;44376 106:     ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                                ;44377         VA_SEXT(XB)+R[GPR.R]_PC_PC+I,  ; ADDRESS OF OPERAND IS REG + DSPLMT
                                ;44378         SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 0106, 0087,7815,003C,C4A7,0049,4              ;44379         NEXT/OS.RED-READ.SAV.EXIT      ; VA GETS ADDRESS OF OPERAND
                                ;44380
                                ;44381 107:     ;0111-------------------------; @#Addr     ABSOLUTE MODE
                                ;44382         VA_XB_PC_PC+4,                 ; NEXT 4 BYTES OF I-STREAM IS OPERAND
                                ;44383         CLOBBER MTEMP0 DEF,            ; PUT GARBAGE IN MTEMP0
U 0107, 0C87,7002,402D,A4A7,0049,4              ;44384         NEXT/OS.RED-READ.SAV.EXIT      ; VA GETS ADDRESS OF OPERAND
```

```
                                ;44385 108:     ;1000----------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;44386          FPA_Q_M[MDR] VA_R[GPR.R],        ; OPERAND ADDRESS IS GPR(RNUM)
                                ;44387          CLOBBER MTEMP0 DEF,              ; PUT GARBAGE IN MTEMP0
U 0108, 0887,2004,703C,C4A7,1010,E ;44388       NEXT/OS.RED-READ.EXIT            ;
                                ;44389
                                ;44390 109:     ;1001----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;44391          VA_PC+SEXT(XB)+I PC_PC+I,        ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;44392          SIZE[IDEP],CLOBBER MTEMP0 DEF,   ; PUT GARBAGE IN MTEMP0
U 0109, 0C87,7816,403D,8407,0049,6 ;44393       NEXT/OS.RED-READ.SAV             ;
                                ;44394
                                ;44395 10A:     ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;44396          VA_SEXT(XB)+R[GPR.R] PC_PC+I,    ; OPERAND IS REG + DATA FROM I-STREAM
                                ;44397          SIZE[IDEP],CLOBBER MTEMP0 DEF,   ; PUT GARBAGE IN MTEMP0
U 010A, 0887,7815,003C,C4A7,0049,6 ;44398       NEXT/OS.RED-READ.SAV             ;
                                ;44399
                                ;44400 10B:     ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;44401          FPA_Q_M[MDR] VA_R[GPR.R],        ; VA GETS OPERAND    GPR(RNUM)
                                ;44402          CLOBBER MTEMP0 DEF,              ; PUT GARBAGE IN MTEMP0
U 010B, 0887,2004,703C,C4A7,104B,A ;44403       NEXT/OS.RED-READ.INC.4           ; SAVE MDR IN Q
                                ;44404
                                ;44405 10C:     ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 010C, 0C80,0036,4030,0047,00FD,C ;44406       NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44407
                                ;44408 10D:     ;1101----------------------------; (Rn)[Rx]  INDEX MODE
                                ;44409          FPA_Q_MDR MTEMP0_R[GPR.R],       ; SAVE INDEX REGISTER IN TEMP0
U 010D, 0C87,2004,71FC,C047,1440,0 ;44410       PUSH,BUT/LOD.BRA,NEXT/OS.BOA     ; CALL ROUTINE TO GET BASE OPERAND ADD
                                ;44411
                                ;44412 10E:
                                ;44413 OS.RED-READ.EXIT:
                                ;44414          ;1110----------------------------;
                                ;44415          READ,SIZE[IDEP],                 ; READ MEMORY AT VA  SIZE=DSIZE
U 010E, 0880,0036,4070,0050,0000,1 ;44416       IRDX [1]                         ;
                                ;44417
                                ;44418
                                ;44419
                                ;44420
                                ;44421 OS.RED-AUTO.INC:
                                ;44422          ;--------------------------------;
                                ;44423          READ,                            ; READ MEMORY ADDRESS VA AND SIZE=DSIZE
                                ;44424          R[GPR.R]_RB+CONX.SIZ,            ; GPR(RNUM) <- GPR(RNUM)+SIZE
                                ;44425          PUSH RBS+,                       ; SAVE STATUS OF GPR(RNUM) ON RBS
U 048C, 0C85,573D,007C,C050,0000,1 ;44426       IRDX [1]                         ;
                                ;44427
                                ;44428 OS.RED-IMMED:
                                ;44429          ;--------------------------------;
                                ;44430          FPA_Q_M[MDR] MDR_Q,              ; SAVE MDR INTO Q
                                ;44431          CLOBBER MTEMP0 DEF,              ; PUT OPERAND INTO MDR
U 048E, 0087,200C,7070,0467,1000,1 ;44432       IRDX [1]                         ;
```

```
                                ;44433 OS.RED-READ.SAV.EXIT:
                                ;44434          ;--------------------------------;
                                ;44435          FPA_Q_M[MDR],                    ; SAVE MDR IN Q
                                ;44436          READ,SIZE[IDEP],                 ; MDR GETS MEMORY(VA)
                                ;44437          CLOBBER MTEMP0 DEF,              ; PUT GARBAGE IN MTEMP0
U 0494, 0087,2592,5070,0050,1000,1  ;44438      IRDX [1]                         ;
                                ;44439
                                ;44440 OS.RED-READ.SAV:
                                ;44441          ;--------------------------------;
                                ;44442          FPA_Q_M[MDR],                    ; SAVE MDR IN Q
                                ;44443          SIZE[LONG],READ,                 ; MDR GETS MEMORY(VA)
U 0496, 0081,2592,5020,0050,1049,A  ;44444      NEXT/OS.RED-VA_MDR               ;
                                ;44445
                                ;44446 OS.RED-VA_MDR:
                                ;44447          ;--------------------------------;
                                ;44448          VA_M[MDR],                       ; VA GETS MDR
U 049A, 0081,2002,403D,84A7,0010,E  ;44449      NEXT/OS.RED-READ.EXIT            ;
                                ;44450
                                ;44451 OS.RED-READ.INC.4:
                                ;44452          ;--------------------------------;
                                ;44453          READ,                            ; MDR <- MEMORY(VA)
                                ;44454          R[GPR.R]_RB+CONX(4),              ; GPR(RNUM) <- GPR(RNUM)+4
                                ;44455          PUSH RBS+,                       ; RBS GETS |RNUM|SIZE|+|
U 04BA, 0C85,573D,002C,C050,0049,A  ;44456      NEXT/OS.RED-VA_MDR               ;
```

```
                                ;44457 .TOC "  Native Mode Operand Specifier   : OS.BOA"
                                ;44458
                                ;44459 ;*****************************************************************************
                                ;44460 ;
                                ;44461 ;       SUBROUTINE CALLED BY OPERAND SPECIFIER ROUTINES TO EVALUATE
                                ;44462 ;       THE BASE OPERANS ADDRESS IN INDEX MODE
                                ;44463 ;
                                ;44464 ;*****************************************************************************
                                ;44465
                                ;44466 =0000
                                ;44467 OS.BOA:
                                ;44468          ;0000-------------------------; Rn        REGISTER MODE
U 0400, 0C80,0036,4030,0047,00FD,C ;44469       NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44470
                                ;44471          ;0001-------------------------; (Rn)+     AUTOINCREMENT MODE
                                ;44472          MDR_R[GPR.R],                   ; MDR GETS GPR(RNUM)
U 0401, 0C80,05BE,403C,C467,004B,B ;44473       NEXT/OS.BOA-INC                 ;
                                ;44474
                                ;44475          ;0010-------------------------; I^#cons   IMMEDIATE MODE
                                ;44476          MDR_XB_PC_PC+I,                 ; THROW AWAY ISTREAM DATA
U 0402, 0081,7002,403D,A467,047F,B ;44477       PUSH,NEXT/OS.BOA-INDEX-IMMED    ; SIZE IS FORM ROM
                                ;44478
                                ;44479          ;0011-------------------------; S^#cons   LITERAL MODE
U 0403, 0C80,0036,4030,0047,00FD,C ;44480       NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44481
                                ;44482          ;0100-------------------------; -(Rn)     AUTODECREMENT MODE
                                ;44483          MDR_R[GPR.R]_RB-CONX.SIZ,       ; MDR GETS GPR(RNUM)-SIZE
                                ;44484          PUSH RBS-,                      ; GPR(RNUM) <- GPR(RNUM)-SIZE
U 0404, 0085,473C,003C,C467,0040,E ;44485       NEXT/OS.BOA-EXIT                ; RBS <- !RNUM!SIZE!-!
                                ;44486
                                ;44487          ;0101-------------------------; Addr      RELATIVE MODE
                                ;44488          VA_PC+SEXT(XB)+I_PC_PC+I,       ; VA GETS PC+DISPLACEMENT+SIZE
U 0405, 0481,7816,403D,8407,004B,C ;44489       SIZE[IDEP],NEXT/OS.BOA-RELATIVE ;
                                ;44490
                                ;44491          ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                                ;44492          MDR_SEXT(XB)+R[GPR.R]_PC_PC+I,  ; MDR GETS GPR(RNUM)+DISPLACEMENT
U 0406, 0081,7815,003C,C467,0040,E ;44493       SIZE[IDEP],NEXT/OS.BOA-EXIT     ;
                                ;44494
                                ;44495          ;0111-------------------------; @#Addr    ABSOLUTE MODE
                                ;44496          MDR_XB_PC_PC+4,                 ; MDR GETS ADDRESS
U 0407, 0C81,7002,402D,A467,0040,E ;44497       NEXT/OS.BOA-EXIT                ;
                                ;44498
                                ;44499          ;1000-------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;44500          MDR_R[GPR.R],                   ; MDR GETS CONTENTS OF GPR(RNUM)
U 0408, 0480,05BE,403C,C467,0040,E ;44501       NEXT/OS.BOA-EXIT                ;
                                ;44502
                                ;44503          ;1001-------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;44504          VA_PC+SEXT(XB)+I_PC_PC+I,       ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 0409, 0C81,7816,403D,8407,004B,D ;44505       SIZE[IDEP],NEXT/OS.BOA-READ     ;
```

```
                                ;44506          ;1010-------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;44507          VA_SEXT(XB)+R[GPR.R] PC_PC+I,  ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 040A, 0881,7815,003C,C4A7,004B,D ;44508       SIZE[IDEP],NEXT/OS.BOA-READ    ; PC GETS PC+SIZE
                                ;44509
                                ;44510          ;1011-------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;44511          VA_R[GPR.R],                   ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 040B, 0C80,05BE,403C,C4A7,004B,E ;44512       NEXT/OS.BOA-READ.INC
                                ;44513
                                ;44514          ;1100-------------------------; (Rn)[PC]  INDEX MODE PC
U 040C, 0C80,0036,4030,0047,00FD,C ;44515       NEXT/IE.ADDR MODE              ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44516
                                ;44517          ;1101-------------------------; (Rn)[Rx]  INDEX MODE
U 040D, 0C80,0036,4030,0047,00FD,C ;44518       NEXT/IE.ADDR.MODE              ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44519
                                ;44520 OS.BOA-EXIT:
                                ;44521          ;1110-------------------------;
                                ;44522          VA_M[MDR]+(R[TEMP0],ASL.SIZE), ; ADDRESS IS INDEX REG + BOA
U 040E, 0C81,25D1,00B0,04A7,0000,1 ;44523       SIZE[IDEP],RETURN [1]
                                ;44524 =
                                ;44525 OS.BOA-INC:
                                ;44526          ;-------------------------------;
                                ;44527          R[GPR.R]_RB+CONX.SIZ,           ; GPR(RNUM) <- GPR(RNUM)+SIZE
                                ;44528          PUSH RBS+,                     ; SAVE STATUS OF REGISTER
U 04BB, 0085,573D,003C,C047,0040,F ;44529       NEXT/OS.BOA-EXIT               ;
                                ;44530
                                ;44531 OS.BOA-RELATIVE:
                                ;44532          ;-------------------------------;
                                ;44533          VA_M[VA]+(R[TEMP0].ASL.SIZE),  ; FORM ADDRESS OF BOA
U 04BC, 0C81,B5D1,00B0,04A7,0000,1 ;44534       SIZE[IDEP],RETURN [1]          ;
                                ;44535
                                ;44536 OS.BOA-READ:
                                ;44537          ;-------------------------------;
                                ;44538          READ,SIZE[LONG],               ; MDR <- MEMORY(VA)
U 04BD, 0080,0036,4020,0050,0040,E ;44539       NEXT/OS.BOA-EXIT               ;
                                ;44540
                                ;44541 OS.BOA-READ.INC:
                                ;44542          ;-------------------------------;
                                ;44543          READ,PUSH RBS+,                ; MDR <- MEMORY(VA)
                                ;44544          R[GPR.R]_RB+CONX(4),           ; INC REGISTER BY 4
U 04BE, 0485,573D,002C,C050,0040,E ;44545       NEXT/OS.BOA-EXIT               ;
                                ;44546
                                ;44547 7FB:
                                ;44548 OS.BOA-INDEX-IMMED:
                                ;44549          ;*********FORCE ADDRESS*********;
                                ;44550          MDR_M[PC]-CONX.SIZ,SIZE[IDEP], ; BOA IS UNINCREMENTED PC
U 07FB, 0081,A710,00B0,0467,0000,C ;44551       RETURN [+12.]                  ;
```

```
                        ;44552 .TOC "  Native Mode Operand Specifier   : OS.BOA-WRT1"
                        ;44553
                        ;44554 ;*****************************************************************************
                        ;44555 ;
                        ;44556 ;       SUBROUTINE CALLED BY OPERAND SPECIFIER ROUTINES (OS.WRT1) TO EVALUATE
                        ;44557 ;       THE BASE OPERANS ADDRESS IN INDEX MODE
                        ;44558 ;
                        ;44559 ;*****************************************************************************
                        ;44560
                        ;44561 =0000
                        ;44562 OS.BOA-WRT1:
                        ;44563          ;0000-------------------------; Rn        REGISTER MODE
U 0410, 0C80,0036,4030,0047,00FD,C      ;44564          NEXT/IE.ADDR.MODE             ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                        ;44565
                        ;44566          ;0001-------------------------; (Rn)+      AUTOINCREMENT MODE
                        ;44567          VA_M[TEMP0]+R[GPR.R],         ; FORM BOA
U 0411, 0480,0001,003C,C4A7,004B,F      ;44568          NEXT/OS.BOA-WRT1-INC          ;
                        ;44569
                        ;44570          ;0010-------------------------; !^#cons    IMMEDIATE MODE
                        ;44571          MDR_XB PC_PC+I,               ; THROW AWAY ISTREAM DATA
U 0412, 0081,7002,403D,A467,047F,B      ;44572          PUSH,NEXT/OS.BOA-INDEX-IMMED  ; SIZE IS FORM ROM
                        ;44573
                        ;44574          ;0011-------------------------; S^#cons    LITERAL MODE
U 0413, 0C80,0036,4030,0047,00FD,C      ;44575          NEXT/IE.ADDR.MODE             ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                        ;44576
                        ;44577          ;0100-------------------------; -(Rn)      AUTODECREMENT MODE
                        ;44578          MDR_R[GPR.R]_RB-CONX.SIZ,     ; MDR GETS GPR(RNUM)-SIZE
                        ;44579          PUSH RBS-,                    ; GPR(RNUM) <- GPR(RNUM)-SIZE
U 0414, 0885,473C,003C,C467,0041,E      ;44580          NEXT/OS.BOA-WRT1-EXIT         ; RBS <- !RNUM!SIZE!-!
                        ;44581
                        ;44582          ;0101-------------------------; Addr      RELATIVE MODE
                        ;44583          VA_PC+SEXT(XB)+I PC_PC+I,     ; VA GETS PC+DISPLACEMENT+SIZE
                        ;44584          SIZE[IDEP],                   ;
U 0415, 0C81,7816,403D,8407,004C,0      ;44585          NEXT/OS.BOA-WRT1-RELATIVE     ;
                        ;44586
                        ;44587          ;0110-------------------------; D(Rn)      DISPLACEMENT MODE
                        ;44588          MDR_SEXT(XB)+R[GPR.R] PC_PC+I,; MDR GETS GPR(RNUM)+DISPLACEMENT
U 0416, 0881,7815,003C,C467,0041,E      ;44589          SIZE[IDEP],NEXT/OS.BOA-WRT1-EXIT;
                        ;44590
                        ;44591          ;0111-------------------------; @#Addr     ABSOLUTE MODE
                        ;44592          VA_XB+R[TEMP0] PC_PC+4,       ; GET BASE OPERAND ADDRESS
U 0417, 0481,7001,0060,24A7,0000,1      ;44593          IRDX [1]                      ;
                        ;44594
                        ;44595          ;1000-------------------------; (Rn)       REGISTER DEFERRED MODE
                        ;44596          VA_M[TEMP0]+R[GPR.R],         ; FORM BOA
U 0418, 0880,0001,007C,C4A7,0000,1      ;44597          IRDX [1]                      ;
                        ;44598
                        ;44599          ;1001-------------------------; @Addr      RELATIVE DEFERRED MODE
                        ;44600          VA_PC+SEXT(XB)+I PC_PC+I,     ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 0419, 0481,7816,403D,8407,004C,1      ;44601          SIZE[IDEP],NEXT/OS.BOA-WRT1-READ;
```

```
                                ;44602          ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;44603          VA_SEXT(XB)+R[GPR.R] PC_PC+I,    ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 041A, 0081,7815,003C,C4A7,004C,1 ;44604       SIZE[IDEP],NEXT/OS.BOA-WRT1-READ; PC GETS PC+SIZE
                                ;44605
                                ;44606          ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;44607          VA_R[GPR.R],                     ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 041B, 0480,05BE,403C,C4A7,004C,2 ;44608       NEXT/OS.BOA-WRT1-READ.INC        ;
                                ;44609
                                ;44610          ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 041C, 0C80,0036,4030,0047,00FD,C ;44611       NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44612
                                ;44613          ;1101----------------------------; (Rn)[Rx]  INDEX MODE
U 041D, 0C80,0036,4030,0047,00FD,C ;44614       NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44615
                                ;44616 OS.BOA-WRT1-EXIT:
                                ;44617          ;1110----------------------------;
                                ;44618          VA_M[MDR]+R[TEMP0],              ; ADDRESS IS INDEX REG + BOA
U 041E, 0881,2001,0070,04A7,0000,1 ;44619       SIZE[IDEP],IRDX [1]              ;
                                ;44620 =
                                ;44621 OS.BOA-WRT1-INC:
                                ;44622          ;--------------------------------;
                                ;44623          R[GPR.R] RB+CONX.SIZ,            ; GPR(RNUM) <- GPR(RNUM)+SIZE
                                ;44624          PUSH RBS+,                       ; SAVE STATUS OF REGISTER
U 04BF, 0C85,573D,007C,C047,0000,1 ;44625       IRDX [1]                         ;
                                ;44626
                                ;44627 OS.BOA-WRT1-RELATIVE:
                                ;44628          ;--------------------------------;
                                ;44629          VA_M[VA]+R[TEMP0],               ; FORM ADDRESS OF BOA
U 04C0, 0881,B001,0070,04A7,0000,1 ;44630       IRDx [1]                         ;
                                ;44631
                                ;44632 OS.BOA-WRT1-READ:
                                ;44633          ;--------------------------------;
                                ;44634          READ,SIZE[LONG],                 ; MDR <- MEMORY(VA)
U 04C1, 0880,0036,4020,0050,0041,E ;44635       NEXT/OS.BOA-WRT1-EXIT            ;
                                ;44636
                                ;44637 OS.BOA-WRT1-READ.INC:
                                ;44638          ;--------------------------------;
                                ;44639          READ,PUSH RBS+,                  ; MDR <- MEMORY(VA)
                                ;44640          R[GPR.R] RB+CONX(4),             ; INC REGISTER BY 4
U 04C2, 0C85,573D,002C,C050,0041,E ;44641       NEXT/OS.BOA-WRT1-EXIT            ;
```

```
                        ;44642 .TOC "  Native Mode Operand Specifier   : OS.BOA-WRT2"
                        ;44643
                        ;44644 ;*****************************************************************************
                        ;44645 ;
                        ;44646 ;       SUBROUTINE CALLED BY OPERAND SPECIFIER ROUTINES (OS.WRT2) TO EVALUATE
                        ;44647 ;       THE BASE OPERANDS ADDRESS IN INDEX MODE
                        ;44648 ;
                        ;44649 ;*****************************************************************************
                        ;44650
                        ;44651 =0000
                        ;44652 OS.BOA-WRT2:
                        ;44653         ;0000-------------------------; Rn        REGISTER MODE
U 0420, 0C80,0036,4030,0047,00FD,C      ;44654         NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                        ;44655
                        ;44656         ;0001-------------------------; (Rn)+     AUTOINCREMENT MODE
                        ;44657         MDR_R[GPR.R],                   ; MDR GETS GPR(RNUM)
U 0421, 0480,05BE,403C,C467,004C,4      ;44658         NEXT/OS.BOA-WRT2-INC                    ;
                        ;44659
                        ;44660         ;0010-------------------------; I^#cons   IMMEDIATE MODE
                        ;44661         MDR_XB_PC_PC+I,                 ; THROW AWAY ISTREAM DATA
U 0422, 0081,7002,403D,A467,047F,B      ;44662         PUSH,NEXT/OS.BOA-INDEX-IMMED    ; SIZE IS FROM ROM
                        ;44663
                        ;44664         ;0011-------------------------; S^#cons   LITERAL MODE
U 0423, 0C80,0036,4030,0047,00FD,C      ;44665         NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                        ;44666
                        ;44667         ;0100-------------------------; -(Rn)     AUTODECREMENT MODE
                        ;44668         MDR_R[GPR.R]_RB-CONX.SIZ,       ; MDR GETS GPR(RNUM)-SIZE
                        ;44669         PUSH RBS-,                      ; GPR(RNUM) <- GPR(RNUM)-SIZE
U 0424, 0885,473C,003C,C467,0042,E      ;44670         NEXT/OS.BOA-WRT2-EXIT           ; RBS <- |RNUM|SIZE|-|
                        ;44671
                        ;44672         ;0101-------------------------; Addr      RELATIVE MODE
                        ;44673         VA_PC+SEXT(XB)+I_PC_PC+I,       ; VA GETS PC+DISPLACEMENT+SIZE
                        ;44674         SIZE[IDEP],                     ;
U 0425, 0C81,7816,403D,8407,004C,3      ;44675         NEXT/OS.BOA-WRT2-RELATIVE       ;
                        ;44676
                        ;44677         ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                        ;44678         MDR_SEXT(XB)+R[GPR.R]_PC_PC+I,  ; MDR GETS GPR(RNUM)+DISPLACEMENT
U 0426, 0881,7815,003C,C467,0042,E      ;44679         SIZE[IDEP],NEXT/OS.BOA-WRT2-EXIT;
                        ;44680
                        ;44681         ;0111-------------------------; @#Addr    ABSOLUTE MODE
                        ;44682         MDR_XB_PC_PC+4,                 ; MDR GETS ADDRESS
U 0427, 0481,7002,402D,A467,0042,E      ;44683         NEXT/OS.BOA-WRT2-EXIT           ;
                        ;44684
                        ;44685         ;1000-------------------------; (Rn)      REGISTER DEFERRED MODE
                        ;44686         MDR_R[GPR.R],                   ; MDR GETS CONTENTS OF GPR(RNUM)
U 0428, 0C80,05BE,403C,C467,0042,E      ;44687         NEXT/OS.BOA-WRT2-EXIT           ;
                        ;44688
                        ;44689         ;1001-------------------------; @Addr     RELATIVE DEFERRED MODE
                        ;44690         VA_PC+SEXT(XB)+I_PC_PC+I,       ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 0429, 0C81,7816,403D,8407,004C,5      ;44691         SIZE[IDEP],NEXT/OS.BOA-WRT2-READ;
```

```
                                ;44692          ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;44693          VA_SEXT(XB)+R[GPR.R] PC_PC+I,    ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 042A, 0881,7815,003C,C4A7,004C,5 ;44694       SIZE[IDEP],NEXT/OS.BOA-WRT2-READ; PC GETS PC+SIZE
                                ;44695
                                ;44696          ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;44697          VA_R[GPR.R],                     ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 042B, 0C80,05BE,403C,C4A7,004C,6 ;44698       NEXT/OS.BOA-WRT2-READ.INC
                                ;44699
                                ;44700          ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 042C, 0C80,0036,4030,0047,00FD,C ;44701       NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44702
                                ;44703          ;1101----------------------------; (Rn)[Rx]  INDEX MODE
U 042D, 0C80,0036,4030,0047,00FD,C ;44704       NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44705
                                ;44706 OS.BOA-WRT2-EXIT:
                                ;44707          ;1110----------------------------;
                                ;44708          VA_M[MDR]+(R[TEMP0].ASL.SIZE),   ; ADDRESS IS INDEX REG + BOA
U 042E, 0C81,25D1,0070,04A7,0000,1 ;44709       SIZE[IDEP],IRDX [1]              ;
                                ;44710 =
                                ;44711 OS.BOA-WRT2-RELATIVE:
                                ;44712          ;--------------------------------;
                                ;44713          VA_M[VA]+(R[TEMP0].ASL.SIZE),    ; FORM ADDRESS OF BOA
U 04C3, 0C81,B5D1,0070,04A7,0000,1 ;44714       SIZE[IDEP],IRDX [1]              ;
                                ;44715 OS.BOA-WRT2-INC:
                                ;44716          ;--------------------------------;
                                ;44717          R[GPR.R]_RB+CONX.SIZ,            ; GPR(RNUM) <- GPR(RNUM)+SIZE
                                ;44718          PUSH RBS+,                       ; SAVE STATUS OF REGISTER
U 04C4, 0885,573D,003C,C047,0042,E ;44719       NEXT/OS.BOA-WRT2-EXIT            ;
                                ;44720
                                ;44721 OS.BOA-WRT2-READ:
                                ;44722          ;--------------------------------;
                                ;44723          READ,SIZE[LONG],                 ; MDR <- MEMORY(VA)
U 04C5, 0880,0036,4020,0050,0042,E ;44724       NEXT/OS.BOA-WRT2-EXIT            ;
                                ;44725
                                ;44726 OS.BOA-WRT2-READ.INC:
                                ;44727          ;--------------------------------;
                                ;44728          READ,PUSH RBS+,                  ; MDR <- MEMORY(VA)
                                ;44729          R[GPR.R]_RB+CONX(4),             ; INC REGISTER BY 4
U 04C6, 0C85,573D,002C,C050,0042,E ;44730       NEXT/OS.BOA-WRT2-EXIT            ;
```

```
                                ;44731 .TOC "  Native Mode Operand Specifier   : OS.BOA-QUAD"
                                ;44732
                                ;44733 ;*****************************************************************************
                                ;44734 ;
                                ;44735 ;       SUBROUTINE CALLED BY OPERAND SPECIFIER ROUTINES TO EVALUATE
                                ;44736 ;       THE BASE OPERANS ADDRESS IN INDEX MODE
                                ;44737 ;
                                ;44738 ;*****************************************************************************
                                ;44739
                                ;44740 =0000
                                ;44741 OS.BOA-QUAD:
                                ;44742          ;0000-------------------------; Rn        REGISTER MODE
U 0430, 0C80,0036,4030,0047,00FD,C ;44743       NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44744
                                ;44745          ;0001-------------------------; (Rn)+     AUTOINCREMENT MODE
                                ;44746          MDR_R[GPR.R],                   ; MDR GETS GPR(RNUM)
U 0431, 0C80,05BE,403C,C467,004B,B ;44747       NEXT/OS.BOA-INC                 ;
                                ;44748
                                ;44749          ;0010-------------------------; I^#cons   IMMEDIATE MODE
                                ;44750          PC_M[PC]+ZLIT0[8],              ; IGNORE ALL IMMEDIATE DATA
U 0432, 0181,AC11,0030,4487,00BE,F ;44751       NEXT/OS.BOA-EXIT.QUAD           ; SIZE IS FROM ROM
                                ;44752
                                ;44753          ;0011-------------------------; S^#cons   LITERAL MODE
U 0433, 0C80,0036,4030,0047,00FD,C ;44754       NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44755
                                ;44756          ;0100-------------------------; -(Rn)     AUTODECREMENT MODE
                                ;44757          MDR_R[GPR.R]_RB-CONX.SIZ,       ; MDR GETS GPR(RNUM)-SIZE
                                ;44758          PUSH RBS-,                      ; GPR(RNUM) <- GPR(RNUM)-SIZE
U 0434, 0085,473C,003C,C467,0040,E ;44759       NEXT/OS.BOA-EXIT                ; RBS <- |RNUM|SIZE|-|
                                ;44760
                                ;44761          ;0101-------------------------; Addr      RELATIVE MODE
                                ;44762          VA_PC+SEXT(XB)+I PC_PC+I,       ; VA GETS PC+DISPLACEMENT+SIZE
U 0435, 0481,7816,403D,8407,004B,C ;44763       SIZE[IDEP],NEXT/OS.BOA-RELATIVE ;
                                ;44764
                                ;44765          ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                                ;44766          MDR_SEXT(XB)+R[GPR.R] PC_PC+I,  ; MDR GETS GPR(RNUM)+DISPLACEMENT
U 0436, 0081,7815,003C,C467,0040,E ;44767       SIZE[IDEP],NEXT/OS.BOA-EXIT     ;
                                ;44768
                                ;44769          ;0111-------------------------; @#Addr    ABSOLUTE MODE
                                ;44770          MDR_XB PC_PC+4,                 ; MDR GETS ADDRESS
U 0437, 0C81,7002,402D,A467,0040,E ;44771       NEXT/OS.BOA-EXIT                ;
                                ;44772
                                ;44773          ;1000-------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;44774          MDR_R[GPR.R],                   ; MDR GETS CONTENTS OF GPR(RNUM)
U 0438, 0480,05BE,403C,C467,0040,E ;44775       NEXT/OS.BOA-EXIT                ;
                                ;44776
                                ;44777          ;1001-------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;44778          VA_PC+SEXT(XB)+I PC_PC+I,       ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 0439, 0C81,7816,403D,8407,004B,D ;44779       SIZE[IDEP],NEXT/OS.BOA-READ     ;
```

```
                                ;44780          ;1010--------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;44781          VA_SEXT(XB)+R[GPR.R] PC_PC+I,   ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 043A, 0881,7815,003C,C4A7,004B,D  ;44782      SIZE[IDEP],NEXT/OS.BOA-READ     ; PC GETS PC+SIZE
                                ;44783
                                ;44784          ;1011--------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;44785          VA_R[GPR.R],                    ; VA GETS ADDRESS OF ADDRESS OF OPERAND
U 043B, 0C80,05BE,403C,C4A7,004B,E  ;44786      NEXT/OS.BOA-READ.INC
                                ;44787
                                ;44788          ;1100--------------------------; (Rn)[PC]  INDEX MODE PC
U 043C, 0C80,0036,4030,0047,00FD,C  ;44789      NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44790
                                ;44791          ;1101--------------------------; (Rn)[Rx]  INDEX MODE
U 043D, 0C80,0036,4030,0047,00FD,C  ;44792      NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44793
                                ;44794 =
                                ;44795 0BEF:    ;*********FORCE ADDRESS*********;
                                ;44796 OS.BOA-EXIT.QUAD:
                                ;44797          ;-------------------------------;
                                ;44798          MDR_M[PC]-CONX.SIZ,SIZE[IDEP],  ; BOA IS UNINCREMENTED PC
U 0BEF, 0C81,A710,0030,0467,0042,E  ;44799      NEXT/OS.BOA-WRT2-EXIT           ;
                                ;44800
                                ;44801 04C7:
                                ;44802 FREE.04C7:
                                ;44803          ;-------------------------------;
                                ;44804          VA_R[TEMP0].ASL.SIZE,           ;
U 04C7, 0880,05F7,00B0,04A7,0000,1  ;44805      SIZE[IDEP],RETURN [1]           ;
```

```
                                ;44806 .TOC "  Native Mode Operand Specifier   : OS.MOD"
                                ;44807
                                ;44808
                                ;44809 ;*****************************************  ******************************
                                ;44810 ;       MDR is saved in the Q-register and is always destroyed.
                                ;44811 ;       Evaluates read type operands and places the result in MDR.
                                ;44812 ;       MTEMP0 is destroyed.
                                ;44813 ;       Read access for the operand is checked.
                                ;44814 ;*************************************************************************
                                ;44815
                                ;44816 110:
                                ;44817 OS.MOD:
                                ;44818          ;0000--------------------------; Rn        REGISTER MODE
                                ;44819          FPA_Q_M[MDR] MDR_R[GPR.R],      ; PLACE OPERAND (GPR(RNUM)) IN MDR
                                ;44820          CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0110, 0487,2004,707C,C467,1000,1  ;44821      IRDX [1]                        ; SAVE MDR IN Q BEFORE CLOBBERING IT
                                ;44822
                                ;44823 111:     ;0001--------------------------; (Rn)+      AUTOINCREMENT MODE
                                ;44824          FPA_Q_M[MDR] VA_R[GPR.R],       ; PLACE ADDRESS OF OPERAND IN VA
                                ;44825          CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0111, 0887,2004,703C,C4A7,104C,8  ;44826      NEXT/OS.MOD-AUTO.INC            ; INC GPR(RNUM) BY SIZE
                                ;44827
                                ;44828 112:     ;0010--------------------------; I^#cons   IMMEDIATE MODE
U 0112, 0C80,0036,4030,0047,00FD,C  ;44829      NEXT/IE.ADDR.MODE               ; UNPREDICTABLE, TAKE ADDRESS MODE FAULT
                                ;44830
                                ;44831 113:     ;0011--------------------------; S^#cons   LITERAL MODE
U 0113, 0C80,0036,4030,0047,00FD,C  ;44832      NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44833
                                ;44834 114:     ;0100--------------------------; -(Rn)      AUTODECREMENT MODE
                                ;44835          VA_R[GPR.R]_RB-CONX.SIZ,        ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
                                ;44836          PUSH RBS-,                      ; VA GETS OPERAND ADDRESS
U 0114, 0885,473C,003C,C4A7,004C,9  ;44837      NEXT/OS.MOD-READ.SAV.EXIT       ; RBS GETS !RNUM!SIZE!-!
                                ;44838
                                ;44839 115:     ;0101--------------------------; Addr      RELATIVE MODE
                                ;44840          VA_PC+SEXT(XB)+I_PC_PC+I,       ; ADDRESS OF OPERAND IS PC + DSPLMT
                                ;44841          SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0115, 0C87,7816,403D,8407,004C,9  ;44842      NEXT/OS.MOD-READ.SAV.EXIT       ;
                                ;44843
                                ;44844 116:     ;0110--------------------------; D(Rn)      DISPLACEMENT MODE
                                ;44845          VA_SEXT(XB)+R[GPR.R] PC_PC+I,   ; ADDRESS OF OPERAND IS REG + DSPLMT
                                ;44846          SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0116, 0887,7815,003C,C4A7,004C,9  ;44847      NEXT/OS.MOD-READ.SAV.EXIT       ; VA GETS ADDRESS OF OPERAND
                                ;44848
                                ;44849 117:     ;0111--------------------------; @#Addr     ABSOLUTE MODE
                                ;44850          VA_XB PC PC+4,                  ; NEXT 4 BYTES OF I-STREAM IS OPERAND
                                ;44851          CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0117, 0487,7002,402D,A4A7,004C,9  ;44852      NEXT/OS.MOD-READ.SAV.EXIT       ; VA GETS ADDRESS OF OPERAND
                                ;44853
                                ;44854 118:     ;1000--------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;44855          FPA_Q_M[MDR] VA_R[GPR.R],       ; OPERAND ADDRESS IS GPR(RNUM)
                                ;44856          CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0118, 0087,2004,703C,C4A7,1011,E  ;44857      NEXT/OS.MOD-READ.EXIT           ;
```

```
                                ;44858 119:     ;1001----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;44859          VA_PC+SEXT(XB)+I PC_PC+I,        ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;44860          SIZE[IDEP],CLOBBER MTEMPO DEF,   ; PUT GARBAGE IN MTEMPO
U 0119, 0C87,7816,403D,8407,004C,A ;44861       NEXT/OS.MOD-READ.SAV             ;
                                ;44862
                                ;44863 11A:     ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;44864          VA_SEXT(XB)+R[GPR.R] PC_PC+I,    ; OPERAND IS REG + DATA FROM I-STREAM
                                ;44865          SIZE[IDEP],CLOBBER MTEMPO DEF,   ; PUT GARBAGE IN MTEMPO
U 011A, 0887,7815,003C,C4A7,004C,A ;44866       NEXT/OS.MOD-READ.SAV             ;
                                ;44867
                                ;44868 11B:     ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;44869          FPA_Q_M[MDR] VA_R[GPR.R],        ; VA GETS OPERAND    GPR(RNUM)
                                ;44870          CLOBBER MTEMPO DEF,              ; PUT GARBAGE IN MTEMPO
U 011B, 0087,2004,703C,C4A7,104C,C ;44871       NEXT/OS.MOD-READ.INC.4           ; SAVE MDR IN Q
                                ;44872
                                ;44873 11C:     ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 011C, 0C80,0036,4030,0047,0CFD,C ;44874       NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44875
                                ;44876 11D:     ;1101----------------------------; (Rn)[Rx]  INDEX MODE
                                ;44877          FPA_Q_MDR MTEMPO_R[GPR.R],       ;SAVE INDEX REGISTER IN TEMPO
U 011D, 0C87,2004,71FC,C047,1440,0 ;44878       PUSH,BUT/LOD.BRA,NEXT/OS.BOA     ; CALL ROUTINE TO GET BASE OPERAND ADD
                                ;44879
                                ;44880 11E:
                                ;44881 OS.MOD-READ.EXIT:
                                ;44882          ;1110----------------------------;
                                ;44883          READ.MOD,SIZE[IDEP],             ; READ MEMORY AT VA  SIZE=DSIZE
U 011E, 0080,0036,4070,0054,0000,1 ;44884       IRDX [1]                         ;
                                ;44885
                                ;44886 OS.MOD-AUTO.INC:
                                ;44887          ;--------------------------------;
                                ;44888          READ.MOD,SIZE[IDEP],             ; READ MEMORY ADDRESS VA AND SIZE=DSIZE
                                ;44889          R[GPR.R]_RB+CONX.SIZ,            ; GPR(RNUM) <- GPR(RNUM)+SIZE
                                ;44890          PUSH RBS+,                       ; SAVE STATUS OF GPR(RNUM) ON RBS
U 04C8, 0485,573D,007C,C054,0000,1 ;44891       IRDX [1]                         ;
                                ;44892
                                ;44893 OS.MOD-READ.SAV.EXIT:
                                ;44894          ;--------------------------------;
                                ;44895          FPA_Q_M[MDR],                    ; SAVE MDR IN Q
                                ;44896          READ.MOD,SIZE[IDEP],             ; MDR GETS MEMORY(VA)
                                ;44897          CLOBBER MTEMPO DEF,              ; PUT GARBAGE IN MTEMPO
U 04C9, 0887,2592,5070,0054,1000,1 ;44898       IRDX [1]                         ;
                                ;44899
                                ;44900 OS.MOD-READ.SAV:
                                ;44901          ;--------------------------------;
                                ;44902          FPA_Q_M[MDR],                    ; SAVE MDR IN Q
                                ;44903          SIZE[LONG],READ,                 ; MDR GETS MEMORY(VA)
U 04CA, 0881,2592,5020,0050,104C,B ;44904       NEXT/OS.MOD-VA_MDR               ;
                                ;44905
                                ;44906 OS.MOD-VA_MDR:
                                ;44907          ;--------------------------------;
                                ;44908          VA_M[MDR],                       ; VA GETS MDR
U 04CB, 0881,2002,403D,84A7,0011,F ;44909       NEXT/OS.MOD-READ.EXIT            ;
```

```
                                ;44910 OS.MOD-READ.INC.4:
                                ;44911          ;--------------------------------;
                                ;44912          READ,                            ; MDR <- MEMORY(VA)
                                ;44913          R[GPR.R]_RB+CONX(4),              ; GPR(RNUM) <- GPR(RNUM)+4
                                ;44914          PUSH RBS+,                       ; RBS GETS |RNUM|SIZE|+1
J 04CC, 0485,573D,002C,C050,004C,B      ;44915  NEXT/OS.MOD-VA_MDR               ;
```

```
                                ;44916 .TOC "  Native Mode Operand Specifier   : OS.ADD"
                                ;44917
                                ;44918 ;*****************************************************************************
                                ;44919 ;       MDR is saved in the Q-register and is always destroyed.
                                ;44920 ;       Evaluates address type operands and leaves the address of the
                                ;44921 ;       operand in MDR.
                                ;44922 ;       MTEMP0 is destroyed.
                                ;44923 ;       No access to the operand is checked.
                                ;44924 ;*****************************************************************************
                                ;44925
                                ;44926 120:
                                ;44927 OS.ADD:
                                ;44928          ;0000-------------------------; Rn        REGISTER MODE
U 0120, 0C80,0036,4030,0047,00FD,C ;44929       NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44930
                                ;44931 121:     ;0001-------------------------; (Rn)+      AUTOINCREMENT MODE
                                ;44932          Q_M[MDR] MDR_R[GPR.R],          ; GPR(RNUM) IS THE ADDRESS OF OPERAND
                                ;44933          C[OBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0121, 0087,2004,703C,C467,004D,3 ;44934       NEXT/OS.ADD-AUTO.INC            ; SAVE MDR IN Q
                                ;44935
                                ;44936 122:     ;0010-------------------------; I^#cons    IMMEDIATE MODE
                                ;44937          Q_M[MDR],                       ; SAVE Q
U 0122, 0481,2592,5030,0047,004C,D ;44938       NEXT/OS.ADD-IMMEDIATE           ;
                                ;44939
                                ;44940 123:     ;0011-------------------------; S^#cons    LITERAL MODE
U 0123, 0C80,0036,4030,0047,00FD,C ;44941       NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44942
                                ;44943 124:     ;0100-------------------------; -(Rn)      AUTODECREMENT MODE
                                ;44944          Q_M[MDR],                       ; SAVE MDR IN Q
                                ;44945          C[OBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0124, 0C87,2592,5030,0047,004C,F ;44946       NEXT/OS.ADD-AUTO.DEC            ;
                                ;44947
                                ;44948 125:     ;0101-------------------------; Addr       RELATIVE MODE
                                ;44949          VA_PC+SEXT(XB)+I PC_PC+I,       ; ADDRESS IS UPDATED PC+DISP
                                ;44950          SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0125, 0487,7816,403D,8407,004D,0 ;44951       NEXT/OS.ADD-RELATIVE            ;
                                ;44952
                                ;44953 126:     ;0110-------------------------; D(Rn)      DISPLACEMENT MODE
                                ;44954          MTEMP0_SEXT(XB)+R[GPR.R] PC_PC+I,; MTEMP0 GETS ADDRESS
                                ;44955          SIZE[IDEP],                     ; INSTRUCTION DETERMINES THE SIZE
U 0126, 0887,7815,003C,C047,004D,1 ;44956       NEXT/OS.ADD-DISPLACEMENT        ;
                                ;44957
                                ;44958 127:     ;0111-------------------------; @#Addr     ABSOLUTE MODE
                                ;44959          MTEMP0_SEXT(XB)+R[ZERO] PC_PC+4,; RTEMP0 GETS ADDRESS, ALWAYS 4 BYTES
U 0127, 0487,7815,002D,A047,004D,1 ;44960       NEXT/OS.ADD-DISPLACEMENT        ; PC <- PC+SIZE
                                ;44961
                                ;44962 128:     ;1000-------------------------; (Rn)       REGISTER DEFERRED MODE
                                ;44963          Q_M[MDR] MDR_R[GPR.R],          ; GPR(RNUM) IS THE OPERAND ADDRESS
                                ;44964          C[OBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0128, 0C87,2004,707C,C467,0000,1 ;44965       IRDX [1]                        ;
                                ;44966
                                ;44967 129:     ;1001-------------------------; @Addr      RELATIVE DEFERRED MODE
                                ;44968          VA_PC+SEXT(XB)+I PC_PC+I,       ; VA GETS ADDRESS OD ADDRESS OF OPERAND
                                ;44969          SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0129, 0C87,7816,403D,8407,004D,2 ;44970       NEXT/OS.ADD-READ.SAV.EXIT       ;
```

```
                                ;44971 12A:     ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;44972          VA_SEXT(XB)+R[GPR.R] PC_PC+I,    ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;44973          SIZE[IDEP],CLOBBER MTEMPO DEF,   ; PUT GARBAGE IN MTEMPO
U 012A, 0887,7815,003C,C4A7,004D,2 ;44974       NEXT/OS.ADD-READ.SAV.EXIT        ;
                                ;44975
                                ;44976 12B:     ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;44977          Q_M[MDR] VA_R[GPR.R],            ; GPR(RNUM) IS ADD OF ADD OF OPERAND
                                ;44978          CLOBBER MTEMPO DEF,              ; PUT GARBAGE IN MTEMPO
U 012B, 0887,2004,703C,C4A7,004D,4 ;44979       NEXT/OS.ADD-READ.INC.4           ;
                                ;44980
                                ;44981 12C:     ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 012C, 0C80,0036,4030,0047,00FD,C ;44982       NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;44983
                                ;44984 12D:     ;1101----------------------------; (Rn)[Rx]  INDEX MODE
                                ;44985          MTEMPO_R[GPR.R] Q_MDR,           ; SAVE INDEX REGISTER
U 012D, 0487,2004,71FC,C047,0440,0 ;44986       PUSH,BUT/LOD.BRA,NEXT/OS.BOA     ;
                                ;44987
                                ;44988 12E:
                                ;44989 OS.ADD-INDEX:
                                ;44990          ;1110----------------------------;
                                ;44991          MDR_M[VA],                       ; PLACE ADDRESS IN MDR
U 012E, 0C81,B002,407D,8467,0000,1 ;44992       IRDX [1]                         ;
                                ;44993
                                ;44994 OS.ADD-IMMEDIATE:
                                ;44995          ;--------------------------------;
                                ;44996          MDR_M[PC],                       ; PC IS THE ADDRESS OF IMMEDIATE DATA
                                ;44997          CLOBBER MTEMPO DEF,              ; PUT GARBAGE IN MTEMPO
U 04CD, 1487,A002,403D,8467,004C,E ;44998       PATCH,NEXT/OS.ADD-INC.PC         ; *** JUMP TO PCS ***
                                ;44999 28CE:
                                ;45000          ;--------------------------------; *** PCS MICROWORD ***
                                ;45001          PC_M[PC]+CONX.SIZ,               ; PUSH PC PAST IMMEDIATE DATA
U 28CE, 0881,A711,0030,0487,0014,0 ;45002       NEXT/OS.WRT1
                                ;45003
                                ;45004 OS.ADD-INC.PC:
                                ;45005          ;################################; MICROWORD REPLACED IN PCS
U 04CE, 0C81,7036,4030,2047,0014,0 ;45006       PC_PC+I MB_XB,NEXT/OS.WRT1       ; PUSH PC PAST IMMEDIATE DATA
                                ;45007
                                ;45008 OS.ADD-AUTO.DEC:
                                ;45009          ;--------------------------------;
                                ;45010          MDR_R[GPR.R]_RB-CONX.SIZ,        ; ADDRESS IS GPR(RNUM)-SIZE
                                ;45011          PUSH RBS-,                       ; SAVE STATE OF REGISTER
U 04CF, 0C85,473C,007C,C467,0000,1 ;45012       IRDX [1]                         ;
                                ;45013
                                ;45014 OS.ADD-RELATIVE:
                                ;45015          ;--------------------------------;
                                ;45016          Q_M[MDR],                        ; SAVE MDR IN Q
U 04D0, 0C81,2592,5030,0047,0012,E ;45017       NEXT/OS.ADD-INDEX                ;
                                ;45018
                                ;45019 OS.ADD-DISPLACEMENT:
                                ;45020          ;--------------------------------;
                                ;45021          Q_M[MDR] MDR_R[TEMPO],           ; GET ADDRESS FROM TEMPO
U 04D1, 0C81,2004,7070,0467,0000,1 ;45022       IRDX [1]                         ;
```

```
                                ;45023 OS.ADD-READ.SAV.EXIT:
                                ;45024          ;--------------------------------;
                                ;45025          Q_M[MDR],                       ; SAVE ANY FIRST OPERAND
                                ;45026          SIZE[LONG],READ,                ; GET ADDRESS OF OPERAND
U 04D2, 0C81,2592,5060,0050,0000,1      ;45027          IRDX [1]                        ;
                                ;45028
                                ;45029 OS.ADD-AUTO.INC:
                                ;45030          ;--------------------------------;
                                ;45031          R[GPR.R]_RB+CONX.SIZ,            ; INC REGISTER BY SIZE
                                ;45032          PUSH RBS+,                      ;
U 04D3, 0C85,573D,007C,C047,0000,1      ;45033          IRDX [1]                        ;
                                ;45034
                                ;45035 OS.ADD-READ.INC.4:
                                ;45036          ;--------------------------------;
                                ;45037          READ,                           ; GET ADDRESS OPERAND
                                ;45038          R[GPR.R]_RB+CONX(4),             ; ALWAYS BUMP REG BY FOR IN INDIRECT OSR
                                ;45039          PUSH RBS+,                      ; SAVE REG CONTENTS IN CASE OF PROBLEMS
U 04D4, 0885,573D,006C,C050,0000,1      ;45040          IRDX [1]                        ;
```

```
                                ;45041 .TOC "  Native Mode Operand Specifier   : OS.VADD"
                                ;45042
                                ;45043 ;*****************************************************************************
                                ;45044 ;       MDR is not saved and is always destroyed.
                                ;45045 ;       Evaluates address type operands and leaves the address of the
                                ;45046 ;       operand in MDR.
                                ;45047 ;       MTEMP0 is destroyed.
                                ;45048 ;       No access to the operand is checked.
                                ;45049 ;*****************************************************************************
                                ;45050
                                ;45051
                                ;45052
                                ;45053 130:
                                ;45054 OS.VADD:
                                ;45055          ;0000-------------------------; Rn        REGISTER MODE
                                ;45056          WB_R[TEMP1].SR.5(IF MDR IS 0),  ; SEE IF POS IS GT 31. FOR PAUL
                                ;45057          ALUS_SIGND,SIZE[LONG],          ; PAUL HAS SET MDR TO ZERO
U 0130, 0081,2251,8060,58C7,0000,4      ;45058          IRDX [4]                        ; SAVE RESULT FOR LATER
                                ;45059
                                ;45060 131:     ;0001-------------------------; (Rn)+      AUTOINCREMENT MODE
                                ;45061          MDR_R[GPR.R],                   ; GPR(RNUM) IS THE ADDRESS OF OPERAND
                                ;45062          CLOBBER MTEMP0,                 ; PUT GARBAGE IN MTEMP0
U 0131, 0486,05BE,403C,C467,004D,9      ;45063          NEXT/OS.VADD-AUTO.INC           ; SAVE MDR IN Q
                                ;45064
                                ;45065 132:     ;0010-------------------------; I^#cons    IMMEDIATE MODE
                                ;45066          R[TEMP0]_M[PC],                 ; PC IA ADDRESS OF IMMEDIATE DATA
U 0132, 0885,A592,4030,0047,004D,8      ;45067          NEXT/OS.VADD-INC.PC             ;
                                ;45068
                                ;45069 133:     ;0011-------------------------; S^#cons    LITERAL MODE
U 0133, 0C80,0036,4030,0047,00FD,C      ;45070          NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45071
                                ;45072 134:     ;0100-------------------------; -(Rn)      AUTODECREMENT MODE
                                ;45073          MDR_R[GPR.R]_RB-CONX.SIZ,       ; ADDRESS IS GPR - SIZE
                                ;45074          PUSH RBS-,                      ; SAVE STATE OF REGISTER FOR RESTART
U 0134, 0485,473C,007C,C467,0000,3      ;45075          IRDX [3]                        ;
                                ;45076
                                ;45077 135:     ;0101-------------------------; Addr       RELATIVE MODE
                                ;45078          VA_PC+SEXT(XB)+I_PC_PC+I,       ; ADDRESS IS UPDATED PC+DISP
                                ;45079          SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0135, 0487,7816,403D,8407,0013,E      ;45080          NEXT/OS.VADD-INDEX              ;
                                ;45081
                                ;45082 136:     ;0110-------------------------; D(Rn)      DISPLACEMENT MODE
                                ;45083          MTEMP0_SEXT(XB)+R[GPR.R] PC_PC+I,; MTEMP0 GETS ADDRESS
                                ;45084          SIZE[IDEP],                     ; LET INSTRUCTION DETERMINE SIZE
U 0136, 0087,7815,003C,C047,004D,5      ;45085          NEXT/OS.VADD-DISPLACEMENT       ;
                                ;45086
                                ;45087 137:     ;0111-------------------------; @#Addr     ABSOLUTE MODE
                                ;45088          MTEMP0_SEXT(XB)+R[ZERO] PC_PC+4,; RTEMP0 GETS ADDRESS. ALWAYS 4 BYTES
U 0137, 0C87,7815,002D,A047,004D,5      ;45089          NEXT/OS.VADD-DISPLACEMENT       ; PC <- PC+SIZE
                                ;45090
                                ;45091 138:     ;1000-------------------------; (Rn)       REGISTER DEFERRED MODE
                                ;45092          MDR_R[GPR.R],                   ; GPR(RNUM) IS THE OPERAND ADDRESS
                                ;45093          CLOBBER MTEMP0,                 ; PUT GARBAGE IN MTEMP0
U 0138, 0086,05BE,407C,C467,0000,3      ;45094          IRDX [3]                        ;
```

```
                                ;45095 139:    ;1001-----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;45096         VA_PC+SEXT(XB)+I PC_PC+I,        ; VA GETS ADDRESS OD ADDRESS OF OPERAND
                                ;45097         SIZE[IDEP],CLOBBER MTEMPO DEF,   ; PUT GARBAGE IN MTEMPO
U 0139, 0487,7816,403D,8407,004D,6 ;45098      NEXT/OS.VADD-READ.EXIT           ;
                                ;45099
                                ;45100 13A:    ;1010-----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;45101         VA_SEXT(XB)+R[GPR.R] PC_PC+I,    ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;45102         SIZE[IDEP],CLOBBER MTEMPO DEF,   ; PUT GARBAGE IN MTEMPO
U 013A, 0087,7815,003C,C4A7,004D,6 ;45103      NEXT/OS.VADD-READ.EXIT           ;
                                ;45104
                                ;45105 13B:    ;1011-----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;45106         VA_R[GPR.R],                     ; GPR(RNUM) IS ADD OF ADD OF OPERAND
                                ;45107         CLOBBER MTEMPO,                  ; PUT GARBAGE IN MTEMPO
U 013B, 0C86,05BE,403C,C4A7,004D,7 ;45108      NEXT/OS.VADD-READ.INC.4          ;
                                ;45109
                                ;45110 13C:    ;1100-----------------------------; (Rn)[PC]  INDEX MODE PC
U 013C, 0C80,0036,4030,0047,00FD,C ;45111      NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45112
                                ;45113 13D:    ;1101-----------------------------; (Rn)[Rx]  INDEX MODE
                                ;45114         M[TEMPO]_R[GPR.R],               ; SAVE INDEX REGISTER
U 013D, 0086,05BE,41FC,C047,0440,0 ;45115      PUSH,BUT7LOD.BRA,NEXT/OS.BOA     ;
                                ;45116
                                ;45117 13E:
                                ;45118 OS.VADD-INDEX:
                                ;45119         ;1110-----------------------------;
                                ;45120         MDR_M[VA],                       ; PLACE ADDRESS IN MDR
U 013E, 0481,B002,407D,8467,0000,3 ;45121      IRDX [3]                         ;
                                ;45122
                                ;45123 OS.VADD-DISPLACEMENT:
                                ;45124         ;--------------------------------;
                                ;45125         MDR_R[TEMPO],                    ; GET ADDRESS FROM TEMPO
U 04D5, 0080,05BE,4070,0467,0000,3 ;45126      IRDX [3]                         ;
                                ;45127
                                ;45128 OS.VADD-READ.EXIT:
                                ;45129         ;--------------------------------;
                                ;45130         READ,                            ; READ MEMORY AT VA  SIZE=DSIZE
U 04D6, 0480,0036,4060,0050,0000,3 ;45131      SIZE[LONG],IRDX [3]              ;
                                ;45132
                                ;45133 OS.VADD-READ.INC.4:
                                ;45134         ;--------------------------------;
                                ;45135         READ,                            ; MDR <- MEMORY(VA)
                                ;45136         R[GPR.R]_RB+CONX(4),             ; GPR(RNUM) <- GPR(RNUM)+4
                                ;45137         PUSH RBS+,                       ; RBS GETS !RNUM!SIZE!+!
U 04D7, 0085,573D,006C,C050,0000,3 ;45138      IRDX [3]                         ;
                                ;45139
                                ;45140 OS.VADD-INC.PC:
                                ;45141         ;--------------------------------;
                                ;45142         PC_PC+I MB_XB,                   ; PUSH PC PAST IMMEDIATE DATA
U 04D8, 0C81,7036,4030,2047,004D,5 ;45143      NEXT/OS.VADD-DISPLACEMENT        ;
                                ;45144
                                ;45145 OS.VADD-AUTO.INC:
                                ;45146         ;--------------------------------;
                                ;45147         R[GPR.R]_RB+CONX.SIZ,            ; INC REGISTER BY SIZE
                                ;45148         PUSH RBS+,                       ;
U 04D9, 0485,573D,007C,C047,0000,3 ;45149      IRDX [3]                         ;
```

```
                                ;45150 .TOC "  Native Mode Operand Specifier   : OS.WRT1"
                                ;45151
                                ;45152 ;*****************************************************************************
                                ;45153 ;       MDR is not saved and will be destroyed in relative
                                ;45154 ;       deferred, displacement deferred and auto-increment
                                ;45155 ;       deferred addressing modes.
                                ;45156 ;       Evaluates write type operands leaving the address of the
                                ;45157 ;       operand in VA for memory operands and in RNUM for operands
                                ;45158 ;       in GPR's.
                                ;45159 ;       MTEMP0 is destroyed.
                                ;45160 ;       No access to the operand is checked.
                                ;45161 ;*****************************************************************************
                                ;45162
                                ;45163
                                ;45164
                                ;45165 140:
                                ;45166 OS.WRT1:
                                ;45167          ;0000-------------------------; Rn        REGISTER MODE
                                ;45168          M[TEMP0]_ZLIT0[0AA],          ; PUT GARBAGE IN MTEMP0
U 0140, 0186,0C37,0075,5047,0000,1 ;45169       IRDX [1]                      ; JUST RETURN  -  IRD1 PUT REG # IN RNUM
                                ;45170
                                ;45171 141:     ;0001-------------------------; (Rn)+     AUTOINCREMENT MODE
                                ;45172          VA_R[GPR.R],                  ; PLACE ADDRESS OF OPERAND IN VA
                                ;45173          CLOBBER MTEMP0,               ; PUT GARBAGE IN MTEMP0
U 0141, 0486,05BE,403C,C4A7,004D,3 ;45174       NEXT/OS.ADD-AUTO.INC          ;
                                ;45175
                                ;45176 142:     ;0010-------------------------; I^#cons   IMMEDIATE MODE
U 0142, 0C80,0036,4030,0047,00FD,C ;45177       NEXT/IE.ADDR.MODE             ; UNPREDICTABLE, TAKE ADDRESS MODE FAULT
                                ;45178
                                ;45179 143:     ;0011-------------------------; S^#cons   LITERAL MODE
U 0143, 0C80,0036,4030,0047,00FD,C ;45180       NEXT/IE.ADDR.MODE             ; ILLEGAL ADDRESSING MODE
                                ;45181
                                ;45182 144:
                                ;45183 OS.WRT1-AUTO.DEC:
                                ;45184          ;0100-------------------------; -(Rn)     AUTODECREMENT MODE
                                ;45185          VA_R[GPR.R]_RB-CONX.SIZ,      ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
                                ;45186          PUSH RBS-,                    ; VA GETS OPERAND ADDRESS
U 0144, 0C85,473C,007C,C4A7,0000,1 ;45187       IRDX [1]                      ; RBS GETS !RNUM!SIZE!-!
                                ;45188
                                ;45189 145:
                                ;45190 OS.WRT1-RELATIVE:
                                ;45191          ;0101-------------------------; Addr      RELATIVE MODE
                                ;45192          VA_PC+SEXT(XB)+1 PC_PC+1,     ; ADDRESS OF OPERAND IS PC + DSPLMT
                                ;45193          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 0145, 0887,7816,407D,8407,0000,1 ;45194       IRDX [1]                      ;
                                ;45195
                                ;45196 146:
                                ;45197 OS.WRT1-DISPLACEMENT:
                                ;45198          ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                                ;45199          VA_SEXT(XB)+R[GPR.R] PC_PC+1, ; ADDRESS OF OPERAND IS REG + DSPLMT
                                ;45200          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 0146, 0C87,7815,007C,C4A7,0000,1 ;45201       IRDX [1]                      ; VA GETS ADDRESS OF OPERAND
```

```
                                ;45202 147:
                                ;45203 OS.WRT1-ABSOLUTE:
                                ;45204          ;0111----------------------------; @#Addr    ABSOLUTE MODE
                                ;45205          VA_XB_PC_PC+4,                   ; NEXT 4 BYTES OF I-STREAM IS OPERAND
                                ;45206          CLOBBER MTEMPO DEF,              ; PUT GARBAGE IN MTEMPO
U 0147, 0087,7002,406D,A4A7,0000,1 ;45207       IRDX [1]                         ; VA GETS ADDRESS OF OPERAND
                                ;45208
                                ;45209 148:     ;1000----------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;45210          VA_R[GPR.R],                     ; OPERAND ADDRESS IS GPR(RNUM)
                                ;45211          CLOBBER MTEMPO,                  ; PUT GARBAGE IN MTEMPO
U 0148, 0886,05BE,407C,C4A7,0000,1 ;45212       IRDX [1]                         ;
                                ;45213
                                ;45214 149:     ;1001----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;45215          VA_PC+SEXT(XB)+1 PC_PC+1,        ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;45216          SIZE[IDEP],CLOBBER MTEMPO DEF,   ; PUT GARBAGE IN MTEMPO
U 0149, 0487,7816,403D,8407,004D,A ;45217       NEXT/OS.WRT1-READ                ;
                                ;45218
                                ;45219 14A:     ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;45220          VA_SEXT(XB)+R[GPR.R] PC_PC+1,    ; OPERAND IS REG + DATA FROM I-STREAM
                                ;45221          SIZE[IDEP],CLOBBER MTEMPO DEF,   ; PUT GARBAGE IN MTEMPO
U 014A, 0087,7815,003C,C4A7,004D,A ;45222       NEXT/OS.WRT1-READ                ;
                                ;45223
                                ;45224 14B:     ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;45225          VA_R[GPR.R],                     ; VA GETS OPERAND    GPR(RNUM)
                                ;45226          CLOBBER MTEMPO,                  ; PUT GARBAGE IN MTEMPO
U 014B, 0C86,05BE,403C,C4A7,004D,B ;45227       NEXT/OS.WRT1-READ.INC.4          ; SAVE MDR IN Q
                                ;45228
                                ;45229 14C:     ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 014C, 0C80,0036,4030,0047,00FD,C ;45230       NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45231
                                ;45232 14D:     ;1101----------------------------; (Rn)[Px]  INDEX MODE
                                ;45233          M[TEMPO] (R[GPR.R].ASL.SIZE),    ; SAVE INDEX REGISTER IN TEMPO
                                ;45234          SIZE[IDEP],                      ;
U 014D, 0086,05F7,01FC,C047,0041,0 ;45235       BUT/LOD.BRA,NEXT/OS.BOA-WRT1     ; GO TO ROUTINE TO GET BOA
                                ;45236
                                ;45237 OS.WRT1-READ:
                                ;45238          ;--------------------------------;
                                ;45239          READ,SIZE[LONG],                 ; MDR GETS MEMORY(VA)
U 04DA, 0080,0036,4020,0050,004D,C ;45240       NEXT/OS.WRT1-DEFERRED            ;
                                ;45241
                                ;45242 OS.WRT1-READ.INC.4:
                                ;45243          ;--------------------------------;
                                ;45244          READ,                            ; MDR <- MEMORY(VA)
                                ;45245          R[GPR.R]_RB+CONX(4),              ; GPR(RNUM) <- GPR(RNUM)+4
                                ;45246          PUSH PBS+,                       ; RBS GETS !RNUM!SIZE!+1
U 04DB, 0485,573D,0C2C,C050,004D,C ;45247       NEXT/OS.WRT1-DEFERRED            ;
                                ;45248
                                ;45249 OS.WRT1-DEFERRED:
                                ;45250          ;--------------------------------;
                                ;45251          VA_M[MDR],                       ; PUT ADDRESS IN VA
U 04DC, 0C81,2002,407D,84A7,0000,1 ;45252       IRDX [1]                         ;
```

```
                        ;45253 .TOC "  Native Mode Operand Specifier   : OS.WRT2"
                        ;45254
                        ;45255 ;*****************************************************************************
                        ;45256 ;       MDR is saved in the Q-register and is destroyed in
                        ;45257 ;       relative deferred, displacement deferred and in
                        ;45258 ;       auto-increment deferred addressing modes.
                        ;45259 ;       Evaluates write type operands leaving the address of the
                        ;45260 ;       operand in VA for operands in memory and in RNUM for
                        ;45261 ;       operands in GPR's.
                        ;45262 ;       OS.WRT2 is one cycle longer than OS.WRT1 for
                        ;45263 ;       auto-decrement, relative, absolute and displacement addressing
                        ;45264 ;       modes.
                        ;45265 ;       MTEMP0 is destroyed.
                        ;45266 ;       No access to the operand is checked.
                        ;45267 ;*****************************************************************************
                        ;45268
                        ;45269
                        ;45270
                        ;45271 150:
                        ;45272 OS.WRT2:
                        ;45273          ;0000-----------------------------; Rn        REGISTER MODE
                        ;45274          Q_M[MDR],                         ; SAVE MDR IN Q
                        ;45275          C[OBBER MTEMP0 DEF,               ; PUT GARBAGE IN MTEMP0
U 0150, 0887,2592,5070,0047,0000,1      ;45276          IRDX [1]                          ; RNUM HAS NUMBER OF REGISTER TO WRITE
                        ;45277
                        ;45278 151:     ;0001-----------------------------; (Rn)+     AUTOINCREMENT MODE
                        ;45279          Q_M[MDR] VA_R[GPR.R],             ; GPR(RNUM) IS THE ADDRESS OF OPERAND
                        ;45280          C[OBBER MTEMP0 DEF,               ; PUT GARBAGE IN MTEMP0
U 0151, 0087,2004,703C,C4A7,004D,3      ;45281          NEXT/OS.ADD-AUTO.INC              ; SAVE MDR IN Q
                        ;45282
                        ;45283 152:     ;0010-----------------------------; I^#cons   IMMEDIATE MODE
U 0152, 0C80,0036,4030,0047,00FD,C      ;45284          NEXT/IE.ADDR.MODE                 ; UNPREDICTABLE, TAKE ADDRESS MODE FAULT
                        ;45285
                        ;45286 153:     ;0011-----------------------------; S^#cons   LITERAL MODE
U 0153, 0C80,0036,4030,0047,00FD,C      ;45287          NEXT/IE.ADDR.MODE                 ; ILLEGAL ADDRESSING MODE
                        ;45288
                        ;45289 154:     ;0100-----------------------------; -(Rn)     AUTODECREMENT MODE
                        ;45290          VA_R[GPR.R]_RB-CONX.SIZ,          ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
                        ;45291          PUSH RBS-,                        ; VA GETS OPERAND ADDRESS
U 0154, 0885,473C,003C,C4A7,0015,0      ;45292          NEXT/OS.WRT2                      ; SAVE MDR IN Q BEFORE EXITING
                        ;45293
                        ;45294 155:     ;0101-----------------------------; Addr      RELATIVE MODE
                        ;45295          VA_PC+SEXT(XB)+I PC_PC+I,         ; ADDRESS OF OPERAND IS PC + DSPLMT
                        ;45296          SIZE[IDEP],                       ; LET INSTRUCTION DETERMINE SIZE
U 0155, 0C81,7816,403D,8407,0015,0      ;45297          NEXT/OS.WRT2                      ; SAVE MDR IN Q BEFORE EXITING
                        ;45298
                        ;45299 156:     ;0110-----------------------------; D(Rn)     DISPLACEMENT MODE
                        ;45300          VA_SEXT(XB)+R[GPR.R] PC_PC+I,     ; ADDRESS OF OPERAND IS REG + DSPLMT
                        ;45301          SIZE[IDEP],                       ; LET INSTRUCTION DETERMINE SIZE
U 0156, 0881,7815,003C,C4A7,0015,0      ;45302          NEXT/OS.WRT2                      ; SAVE MDR IN Q BEFORE EXITING
                        ;45303
                        ;45304 157:     ;0111-----------------------------; @#Addr     ABSOLUTE MODE
                        ;45305          VA_XB PC_PC+4,                    ; NEXT 4 BYTES OF I-STREAM IS OPERAND
U 0157, 0481,7002,402D,A4A7,0015,0      ;45306          NEXT/OS.WRT2                      ; SAVE MDR IN Q BEFORE EXITING
```

```
                                ;45307 158:    ;1000----------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;45308         Q_M[MDR] VA_R[GPR.R],            ; GPR(RNUM) IS THE OPERAND ADDRESS
                                ;45309         C[OBBER MTEMP0 DEF,              ; PUT GARBAGE IN MTEMP0
U 0158, 0C87,2004,707C,C4A7,0000,1      ;45310         IRDX [1]                         ;
                                ;45311
                                ;45312 159:    ;1001----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;45313         VA_PC+SEXT(XB)+I PC_PC+I,        ; VA GETS ADDRESS OD ADDRESS OF OPERAND
                                ;45314         SIZE[IDEP],CLOBBER MTEMP0 DEF,   ; PUT GARBAGE IN MTEMP0
U 0159, 0C87,7816,403D,8407,004D,D      ;45315         NEXT/OS.WRT2-READ                ;
                                ;45316
                                ;45317 15A:    ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;45318         VA_SEXT(XB)+R[GPR.R] PC_PC+I,    ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;45319         SIZE[IDEP],CLOBBER MTEMP0 DEF,   ; PUT GARBAGE IN MTEMP0
U 015A, 0887,7815,003C,C4A7,004D,D      ;45320         NEXT/OS.WRT2-READ                ;
                                ;45321
                                ;45322 15B:    ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;45323         Q_M[MDR] VA_R[GPR.R],            ; GPR(RNUM) IS ADD OF ADD OF OPERAND
                                ;45324         C[OBBER MTEMP0 DEF,              ; PUT GARBAGE IN MTEMP0
U 015B, 0887,2004,703C,C4A7,004D,B      ;45325         NEXT/OS.WRT1-READ.INC.4          ;
                                ;45326
                                ;45327 15C:    ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 015C, 0C80,0036,4030,0047,00FD,C      ;45328         NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45329
                                ;45330 15D:    ;1101----------------------------; (Rn)[Rx]  INDEX MODE
                                ;45331         MTEMP0_R[GPR.R] Q_MDR,           ; SAVE INDEX REGISTER
U 015D, 0487,2004,71FC,C047,0042,0      ;45332         BUT/LOD.BRA,NEXT/OS.BOA-WRT2     ; GET BOA
                                ;45333
                                ;45334 OS.WRT2-READ:
                                ;45335         ;--------------------------------;
                                ;45336         Q_M[MDR],                        ; SAVE MDR IN Q
                                ;45337         SIZE[LONG],READ,                 ; MDR GETS MEMORY(VA)
U 04DD, 0081,2592,5020,0050,004D,C      ;45338         NEXT/OS.WRT1-DEFERRED            ;
```

```
                        ;45339 .TOC "  Native Mode Operand Specifier   : OS.FRED"
                        ;45340
                        ;45341 ;*****************************************************************************
                        ;45342 ;       MDR is saved in the Q-register and is always destroyed.
                        ;45343 ;       Evaluates read type operands for warm floating point
                        ;45344 ;       instructions and places the operand in MDR.
                        ;45345 ;       MTEMP0 is destroyed.
                        ;45346 ;       Read access is checked.
                        ;45347 ;*****************************************************************************
                        ;45348
                        ;45349
                        ;45350
                        ;45351 160:
                        ;45352 OS.FRED:
                        ;45353          ;0000-------------------------; Rn        REGISTER MODE
                        ;45354          FPA_Q_M[MDR] MDR_R[GPR.R],     ; PLACE OPERAND (GPR(RNUM)) IN MDR
                        ;45355          CLOBBER MTEMP0 DEF,            ; PUT GARBAGE IN MTEMP0
U 0160, 0487,2004,707C,C467,1000,1      ;45356          IRDX [1]                       ; SAVE MDR IN Q BEFORE CLOBBERING IT
                        ;45357
                        ;45358 161:     ;0001-------------------------; (Rn)+      AUTOINCREMENT MODE
                        ;45359          FPA_Q_M[MDR] VA_R[GPR.R],      ; PLACE ADDRESS OF OPERAND IN VA
                        ;45360          CLOBBER MTEMP0 DEF,            ; PUT GARBAGE IN MTEMP0
U 0161, 0887,2004,703C,C4A7,1048,C      ;45361          NEXT/OS.RED-AUTO.INC           ; INC GPR(RNUM) BY SIZE
                        ;45362
                        ;45363 162:     ;0010-------------------------; I^#cons    IMMEDIATE MODE
                        ;45364          Q_XB_PC_PC+I,                  ; GET PROPER AMOUNT OF IMMEDIATE DATA
U 0162, 0481,7002,503D,A047,0048,E      ;45365          NEXT/OS.RED-IMMED              ;
                        ;45366
                        ;45367 163:     ;0011-------------------------; S^#cons    LITERAL MODE
                        ;45368          FPA_Q_M[MDR],MDR_ZEXT(OSR),    ; MDR GETS OPERAND FROM I-STREAM
                        ;45369          CLOBBER MTEMP0 DEF,            ; PUT GARBAGE IN MTEMP0
U 0163, 0487,2592,5030,05E7,104D,E      ;45370          NEXT/OS.FRED-UNPACK            ; SAVE MDR IN Q
                        ;45371
                        ;45372 164:     ;0100-------------------------; -(Rn)      AUTODECREMENT MODE
                        ;45373          VA_R[GPR.R]_RB-CONX.SIZ,       ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
                        ;45374          PUSH RBS-,                     ; VA GETS OPERAND ADDRESS
U 0164, 0085,473C,003C,C4A7,0049,4      ;45375          NEXT/OS.RED-READ.SAV.EXIT      ; RBS GETS !RNUM!SIZE!-!
                        ;45376
                        ;45377 165:     ;0101-------------------------; Addr      RELATIVE MODE
                        ;45378          VA_PC+SEXT(XB)+I_PC_PC+I,      ; ADDRESS OF OPERAND IS PC + DSPLMT
                        ;45379          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 0165, 0487,7816,403D,8407,0049,4      ;45380          NEXT/OS.RED-READ.SAV.EXIT      ;
                        ;45381
                        ;45382 166:     ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                        ;45383          VA_SEXT(XB)+R[GPR.R] PC_PC+I,  ; ADDRESS OF OPERAND IS REG + DSPLMT
                        ;45384          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 0166, 0087,7815,003C,C4A7,0049,4      ;45385          NEXT/OS.RED-READ.SAV.EXIT      ; VA GETS ADDRESS OF OPERAND
                        ;45386
                        ;45387 167:     ;0111-------------------------; @#Addr     ABSOLUTE MODE
                        ;45388          VA_XB_PC_PC+4,                 ; NEXT 4 BYTES OF I-STREAM IS OPERAND
                        ;45389          CLOBBER MTEMP0 DEF,            ; PUT GARBAGE IN MTEMP0
U 0167, 0C87,7002,402D,A4A7,0049,4      ;45390          NEXT/OS.RED-READ.SAV.EXIT      ; VA GETS ADDRESS OF OPERAND
```

```
                                ;45391 168:   ;1000----------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;45392        FPA_Q_M[MDR] VA_R[GPR.R],       ; OPERAND ADDRESS IS GPR(RNUM)
                                ;45393        CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0168, 0887,2004,703C,C4A7,1010,E ;45394     NEXT/OS.RED-READ.EXIT           ;
                                ;45395
                                ;45396 169:   ;1001----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;45397        VA_PC+SEXT(XB)+I PC_PC+I,       ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;45398        SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0169, 0C87,7816,403D,8407,0049,6 ;45399     NEXT/OS.RED-READ.SAV            ;
                                ;45400
                                ;45401 16A:   ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;45402        VA_SEXT(XB)+R[GPR.R] PC_PC+I,   ; OPERAND IS REG + DATA FROM I-STREAM
                                ;45403        SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 016A, 0887,7815,003C,C4A7,0049,6 ;45404     NEXT/OS.RED-READ.SAV            ;
                                ;45405
                                ;45406 16B:   ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;45407        FPA_Q_M[MDR] VA_R[GPR.R],       ; VA GETS OPERAND    GPR(RNUM)
                                ;45408        CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 016B, 0887,2004,703C,C4A7,104B,A ;45409     NEXT/OS.RED-READ.INC.4          ; SAVE MDR IN Q
                                ;45410
                                ;45411 16C:   ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 016C, 0C80,0036,4030,0047,00FD,C ;45412     NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45413
                                ;45414 16D:   ;1101----------------------------; (Rn)[Rx]  INDEX MODE
                                ;45415        FPA_Q_MDR MTEMP0_R[GPR.R],      ; SAVE INDEX REGISTER IN TEMP0
U 016D, 0C87,2004,71FC,C047,1440,0 ;45416     PUSH,BUT/LOD.BRA,NEXT/OS.BOA    ; CALL ROUTINE TO GET BASE OPERAND ADD
                                ;45417
                                ;45418 16E:   ;1110----------------------------;
                                ;45419        READ,SIZE[IDEP],                ; READ MEMORY AT VA  SIZE=DSIZE
U 016E, 0880,0036,4070.0050,0000,1 ;45420     IRDX [1]                        ;
                                ;45421
                                ;45422 OS.FRED-UNPACK:
                                ;45423        ;------------------------------ ---;
                                ;45424        MDR_M[MDR].FPLIT,               ; UNPACK FLOATING POINT LITERAL
U 04DE, 0881,27B7,0070,0467,0000,1 ;45425     IRDX [1]                        ;
```

```
                        ;45426 .IOC '' Native Mode Operand Specifier   : OS QRED''
                        ;45427
                        ;45428 ;*************************************************************************
                        ;45429 ;       MDR is saved in the Q-register and is always destroyed.
                        ;45430 ;       Used to evaluate read type operands for quad data type
                        ;45431 ;       routines.  The least significant half of the data is left
                        ;45432 ;       in TEMP1 and the most significant half of data is left in MDR.
                        ;45433 ;       MTEMP0 is destroyed.
                        ;45434 ;       Read access is checked.
                        ;45435 ;*************************************************************************
                        ;45436
                        ;45437
                        ;45438
                        ;45439 170:
                        ;45440 OS.QRED:
                        ;45441          ;0000-------------------------; Rn        REGISTER MODE
                        ;45442          M[TEMP1]_R[GPR.R],              ; TEMP0 HAS LOW HALF OF OPERAND
U 0170, 0086,15BE,403C,C047,004D,F      ;45443          NEXT/OS.QRED-REG                ;
                        ;45444
                        ;45445 171:     ;0001-------------------------; (Rn)+     AUTOINCREMENT MODE
                        ;45446          Q_M[MDR]_VA_R[GPR.R],           ; ADDRESS OF LOW PART OF OPERAND
                        ;45447          C[OBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0171, 0087,2004,703C,C4A7,004E,0      ;45448          NEXT/OS.QRED-AUTO.INC           ;
                        ;45449
                        ;45450 172:     ;0010-------------------------; I^#cons   IMMEDIATE MODE
                        ;45451          R[TEMP1]_XB_PC_PC+1,            ; TEMP1 GETS FIRST HALF OF DATA
U 0172, 0085,7592,4030,6047,004E,1      ;45452          NEXT/OS.QRED-IMMEDIATE          ;
                        ;45453
                        ;45454 173:     ;0011-------------------------; S^#cons   LITERAL MODE
                        ;45455          Q_M[MDR],MDR_ZEXT(OSR),         ; MDR GETS LOW HALF OF OPERAND
                        ;45456          C[OBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0173, 0C87,2592,5030,05E7,004E,2      ;45457          NEXT/OS.QRED-LITERAL            ; SAVE MDR IN Q
                        ;45458
                        ;45459 174:     ;0100-------------------------; -(Rn)     AUTODECREMENT MODE
                        ;45460          VA_R[GPR.R]_RB-CONX.SIZ,        ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
                        ;45461          PUSH_RBS-,                      ; SAVE STATUS ON RBS
U 0174, 0085,473C,003C,C4A7,004E,3      ;45462          NEXT/OS.QRED-READ.SAV.INC       ; RBS GETS |RNUM|SIZE|-1
                        ;45463
                        ;45464 175:     ;0101-------------------------; Addr      RELATIVE MODE
                        ;45465          VA_PC+SEXT(XB)+1_PC_PC+1,       ; ADDRESS OF OPERAND IS PC + DSPLMT
                        ;45466          SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0175, 0487,7816,403D,8407,004E,3      ;45467          NEXT/OS.QRED-READ.SAV.INC       ;
                        ;45468
                        ;45469 176:     ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                        ;45470          VA_SEXT(XB)+R[GPR.R]_PC_PC+1,   ; ADDRESS OF OPERAND IS REG + DSPLMT
                        ;45471          SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0176, 0087,7815,003C,C4A7,004E,3      ;45472          NEXT/OS.QRED-READ.SAV.INC       ;
                        ;45473
                        ;45474 177:     ;0111-------------------------; @#Addr    ABSOLUTE MODE
                        ;45475          VA_XB_PC_PC+4,                  ; VA GETS ADDRESS OF OPERAND
                        ;45476          CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0177, 0C87,7002,402D,A4A7,004E,3      ;45477          NEXT/OS.QRED-READ.SAV.INC       ;
```

```
                                ;45478 178:    ;1000--------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;45479         VA_R[GPR.R],                  ; OPERAND ADDRESS IS GPR(RNUM)
                                ;45480         CLOBBER MTEMP0,               ; PUT GARBAGE IN MTEMP0
U 0178, 0486,05BE,403C,C4A7,004E,3 ;45481      NEXT/OS.QRED-READ.SAV.INC     ;
                                ;45482
                                ;45483 179:    ;1001--------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;45484         VA_PC+SEXT(XB)+1 PC_PC+1,     ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;45485         SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 0179, 0487,7816,403D,8407,004E,5 ;45486      NEXT/OS.QRED-READ.IND         ;
                                ;45487
                                ;45488 17A:    ;1010--------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;45489         VA_SEXT(XB)+R[GPR.R] PC_PC+1, ; OPERAND IS REG + DATA FROM I-STREAM
                                ;45490         SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 017A, 0087,7815,003C,C4A7,004E,5 ;45491      NEXT/OS.QRED-READ.IND         ;
                                ;45492
                                ;45493 17B:    ;1011--------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;45494         Q_M[MDR] VA_R[GPR.R],         ; VA GETS OPERAND    GPR(RNUM)
                                ;45495         CLOBBER MTEMP0 DEF,           ; PUT GARBAGE IN MTEMP0
U 017B, 0887,2004,703C,C4A7,004E,7 ;45496      NEXT/OS.QRED-READ.INC.4       ; SAVE MDR IN Q
                                ;45497
                                ;45498 17C:    ;1100--------------------------; (Rn)[PC]  INDEX MODE PC
U 017C, 0C80,0036,4030,0047,00FD,C ;45499      NEXT/IE.ADDR.MODE             ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45500
                                ;45501 17D:    ;1101--------------------------; (Rn)[Rx]  INDEX MODE
                                ;45502         MTEMP0_R[GPR.R] Q_MDR,PUSH,   ; SAVE INDEX REGISTER IN TEMP0
U 017D, 0487,2004,71FC,C047,0443,0 ;45503      BUT/LOD,BRA,NEXT/OS.BOA-QUAD  ; CALL ROUTINE TO GET BASE OPERAND ADD
                                ;45504
                                ;45505 17E:
                                ;45506 OS.QRED-READ1:
                                ;45507         ;1110--------------------------;
                                ;45508         READ,SIZE[IDEP],              ; READ FIRST HALF OF OPERAND
                                ;45509         VA_VA+4,                      ; POINT VA TO NEXT HALF OF OPERAND
                                ;45510         CLOBBER MTEMP0 DEF,           ; PUT GARBAGE IN MTEMP0
U 017E, 0486,0036,4030,0450,004E,4 ;45511      NEXT/OS.QRED-READ2            ;
                                ;45512
                                ;45513 OS.QRED-REG:
                                ;45514         ;-----------------------------;
                                ;45515         Q_M[MDR] MDR_R[GPR.R+1],      ; MDR HAS HIGH HALF OF OPERAND
                                ;45516         CLOBBER MTEMP0 DEF,           ; PUT GARBAGE IN MTEMP0
U 04DF, 0C87,2004,707F,C467,0000,1 ;45517      IRDX [1]                      ; MDR IS SAVED IN Q
                                ;45518
                                ;45519 OS.QRED-AUTO.INC:
                                ;45520         ;-----------------------------;
                                ;45521         READ,                         ; MDR GETS LOW HALF OF OPERAND
                                ;45522         VA_VA+4,                      ; SET UP VA FOR SECOND HALF READ
                                ;45523         R[GPR.R]_RB+CONX.SIZ,         ; ADD 8 TO THE REGISTER
                                ;45524         PUSH RBS+,                    ; SAVE STATE OF REGISTER FOR RESTART
U 04E0, 0085,573D,003C,C450,004E,4 ;45525      NEXT/OS.QRED-READ2            ;
                                ;45526
                                ;45527 OS.QRED-IMMEDIATE:
                                ;45528         ;-----------------------------;
                                ;45529         Q_XB_PC_PC+1,                 ; GET PROPER AMOUNT OF IMMEDIATE DATA
U 04E1, 0481,7002,503D,A047,0048,E ;45530      NEXT/OS.RED-IMMED             ;
```

```
                                ;45531 OS.QRED-LITERAL:
                                ;45532          ;--------------------------------;
                                ;45533          R[TEMP1]_M[MDR],                ; TEMP1 GETS LOW HALF OF OPERAND
U 04E2, 0885,2592,4030,4047,004E,8 ;45534       NEXT/OS.QRED-MDR_0              ; CLEAR UPPER HALF OF OPERAND
                                ;45535
                                ;45536 OS.QRED-READ.SAV.INC:
                                ;45537          ;--------------------------------;
                                ;45538          READ,SIZE[IDEP],                ; GET FIRST HALF OF OPERAND FROM MEM
                                ;45539          FPA_Q_M[MDR],                   ; SAVE MDR IN Q
                                ;45540          CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
                                ;45541          VA_VA+4,                        ; POINT VA TO NEXT HALF OF OPERAND
U 04E3, 0C87,2592,5030,0450,104E,4 ;45542       NEXT/OS.QRED-READ2              ;
                                ;45543
                                ;45544 OS.QRED-READ2:
                                ;45545          ;--------------------------------;
                                ;45546          READ,                           ; GET SECOND HALF OF OPERAND TO MDR
                                ;45547          FPA_R[TEMP1].SIZ_M[MDR],        ; PUT FIRST HALF OF OPERAND IN TEMP1
U 04E4, 0083,2592,4060,4050,1000,1 ;45548       SIZE[LONG],IRDX [1]             ;
                                ;45549
                                ;45550 OS.QRED-READ.IND:
                                ;45551          ;--------------------------------;
                                ;45552          READ,SIZE[LONG],                ; GET ADDRESS OF OPERAND
                                ;45553          FPA_Q_M[MDR],                   ; SAVE MDR
U 04E5, 0881,2592,5020,0050,104E,6 ;45554       NEXT/OS.QRED-INDIRECT           ;
                                ;45555
                                ;45556 OS.QRED-INDIRECT:
                                ;45557          ;--------------------------------;
                                ;45558          VA_M[MDR],                      ; PLACE ADDRESS OF OPERAND INTO VA
U 04E6, 0881,2002,403D,84A7,0017,E ;45559       NEXT/OS.QRED-READ1              ;
                                ;45560
                                ;45561 OS.QRED-READ.INC.4:
                                ;45562          ;--------------------------------;
                                ;45563          READ,PUSH RBS+,                 ; GET ADDRESS OF OPERAND
                                ;45564          R[GPR.R]_RB+CONX(4),            ; ADD 4 TO REGISTER
U 04E7, 0485,573D,002C,C050,004E,6 ;45565       NEXT/OS.QRED-INDIRECT           ;
                                ;45566
                                ;45567 OS.QRED-MDR_0:
                                ;45568          ;--------------------------------;
                                ;45569          MDR_0,                          ; CLEAR UPPER HALF OF OPERAND
U 04E8, 0C80,0036,4030,04E7,0014,0 ;45570       NEXT/OS.WRT1                    ; DO IRDX
```

```
                                ;45571 .TOC "  Native Mode Operand Specifier   : OS.DRED"
                                ;45572
                                ;45573 ;*****************************************************************************
                                ;45574 ;       MDR is not saved and is destroyed in all but register
                                ;45575 ;       and immediate modes.
                                ;45576 ;       Evaluates read type operands for double data type
                                ;45577 ;       routines.  Bits 31-0 of the operand are left in TEMP1 and bits
                                ;45578 ;       63-32 are left in MDR.
                                ;45579 ;       MTEMP0 is destroyed.
                                ;45580 ;       Read access is checked.
                                ;45581 ;       For G-Float LITERAL FLAG4 is set
                                ;45582 ;*****************************************************************************
                                ;45583
                                ;45584
                                ;45585
                                ;45586 180:
                                ;45587 OS.DRED:
                                ;45588          ;0000-------------------------; Rn        REGISTER MODE
                                ;45589          M[TEMP1]_R[GPR.R],              ; TEMP0 HAS LOW HALF OF OPERAND
U 0180, 0086,15BE,403C,C047,004E,9 ;45590       NEXT/OS.DRED-REG                ;
                                ;45591
                                ;45592 181:     ;0001-------------------------; (Rn)+     AUTOINCREMENT MODE
                                ;45593          VA_R[GPR.R],                    ; ADDRESS OF LOW PART OF OPERAND
                                ;45594          CLOBBER MTEMP0,                 ; PUT GARBAGE IN MTEMP0
U 0181, 0486,05BE,403C,C4A7,004E,0 ;45595       NEXT/OS.QRED-AUTO.INC           ;
                                ;45596
                                ;45597 182:     ;0010-------------------------; I^#cons   IMMEDIATE MODE
                                ;45598          R[TEMP1]_XB PC_PC+I,            ; TEMP1 GETS FIRST HALF OF DATA
U 0182, 0885,7592,4030,6047,004E,C ;45599       NEXT/OS.DRED-IMMEDIATE          ;
                                ;45600
                                ;45601 183:     ;0011-------------------------; S^#cons   LITERAL MODE
                                ;45602          MDR_ZEXT(OSR),GFLOAT(FLAG4)?,   ; MDR GETS LOW HALF OF OPERAND
                                ;45603          M[TEMP0]_ZLIT0[0],              ; PUT GARBAGE IN MTEMP0
U 0183, 0586,0C37,0470,05E7,004B,8 ;45604       NEXT/OS.DRED-UNPACK             ; SAVE MDR IN Q
                                ;45605
                                ;45606 184:     ;0100-------------------------; -(Rn)     AUTODECREMENT MODE
                                ;45607          VA_R[GPR.R]_RB-CONX.SIZ,        ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
U 0184, 0885,473C,003C,C4A7,0017,E ;45608       PUSH RBS-,NEXT/OS.QRED-READ1    ; RBS GETS !RNUM!SIZE!-!
                                ;45609
                                ;45610 185:     ;0101-------------------------; Addr      RELATIVE MODE
                                ;45611          VA_PC+SEXT(XB)+I PC_PC+I,       ; ADDRESS OF OPERAND IS PC + DSPLMT
                                ;45612          SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0185, 0C87,7816,403D,8407,0017,E ;45613       NEXT/OS.QRED-READ1              ;
                                ;45614
                                ;45615 186:     ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                                ;45616          VA_SEXT(XB)+R[GPR.R] PC_PC+I,   ; ADDRESS OF OPERAND IS REG + DSPLMT
                                ;45617          SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 0186, 0887,7815,003C,C4A7,0017,E ;45618       NEXT/OS.QRED-READ1              ;
                                ;45619
                                ;45620 187:     ;0111-------------------------; @#Addr    ABSOLUTE MODE
                                ;45621          VA_XB PC_PC+4,                  ; VA GETS ADDRESS OF OPERAND
                                ;45622          CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 0187, 0487,7002,402D,A4A7,0017,E ;45623       NEXT/OS.QRED-READ1              ;
```

```
                                ;45624 188:  ;1000--------------------------; (Rn)     REGISTER DEFERRED MODE
                                ;45625       VA_R[GPR.R],                  ; OPERAND ADDRESS IS GPR(RNUM)
                                ;45626       CLOBBER MTEMP0,               ; PUT GARBAGE IN MTEMP0
U 0188, 0C86,05BE,403C,C4A7,0017,E ;45627    NEXT/OS.QRED-READ1            ;
                                ;45628
                                ;45629 189:  ;1001--------------------------; @Addr    RELATIVE DEFERRED MODE
                                ;45630       VA_PC+SEXT(XB)+I PC_PC+I,     ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;45631       SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 0189, 0487,7816,403D,8407,004E,A ;45632    NEXT/OS.DRED-READ             ;
                                ;45633
                                ;45634 18A:  ;1010--------------------------; @D(Rn)   DISPLACEMENT DEFERRED MODE
                                ;45635       VA_SEXT(XB)+R[GPR.R] PC_PC+I, ; OPERAND IS REG + DATA FROM I-STREAM
                                ;45636       SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 018A, 0087,7815,003C,C4A7,004E,A ;45637    NEXT/OS.DRED-READ             ;
                                ;45638
                                ;45639 18B:  ;1011--------------------------; @(Rn)+   AUTO-INCREMENT DEFERRED MODE
                                ;45640       VA_R[GPR.R],                  ; VA GETS OPERAND    GPR(RNUM)
                                ;45641       CLOBBER MTEMP0,               ; PUT GARBAGE IN MTEMP0
U 018B, 0C86,05BE,403C,C4A7,004E,7 ;45642    NEXT/OS.QRED-READ.INC.4       ; SAVE MDR IN Q
                                ;45643
                                ;45644 18C:  ;1100--------------------------; (Rn)[PC]  INDEX MODE PC
U 018C, 0C80,0036,4030,0047,00FD,C ;45645    NEXT/IE.ADDR.MODE             ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45646
                                ;45647 18D:  ;1101--------------------------; (Rn)[Rx]  INDEX MODE
                                ;45648       M[TEMP0]_R[GPR.R],PUSH,       ; SAVE INDEX REGISTER IN TEMP0
U 018D, 0086,05BE,41FC,C047,0443,0 ;45649    BUT/LOD.BRA,NEXT/OS.BOA-QUAD  ; CALL ROUTINE TO GET BASE OPERAND ADD
                                ;45650
                                ;45651 18E:  ;1110--------------------------;
                                ;45652       READ,SIZE[IDEP],              ; READ FIRST HALF OF OPERAND
                                ;45653       VA_VA+4,                      ; POINT VA TO NEXT HALF OF OPERAND
U 018E, 0480,0036,4030,0450,004E,4 ;45654    NEXT/OS.QRED-READ2            ;
```

```
                                ;45655 OS.DRED-REG:
                                ;45656          ;--------------------------------;
                                ;45657          MDR_R[GPR,R+1],                 ; MDR HAS HIGH HALF OF OPERAND
                                ;45658          CLOBBER MTEMP0,                 ; PUT GARBAGE IN MTEMP0
U 04E9, 0886,05BE,407F,C467,0000,1  ;45659          IRDX [1]                        ;
                                ;45660
                                ;45661 OS.DRED-READ:
                                ;45662          ;--------------------------------;
                                ;45663          READ,SIZE[LONG],                ; GET ADDRESS OF ADDRESS OF OPERAND
U 04EA, 0080,0036,4020,0050,004E,6  ;45664          NEXT/OS.QRED-INDIRECT           ;
                                ;45665
                                ;45666 =0
                                ;45667 OS.DRED-UNPACK:
                                ;45668          ;0-------------------------------;
                                ;45669          R[TEMP1]_M[MDR].FPLIT,          ; TEMP1 GETS FLOATING POINT LITERAL
U 04B8, 0C85,27B7,0030,4047,004E,8  ;45670          NEXT/OS.QRED-MDR_0              ; GO ZERO UPPER HALF OF OPERAND
                                ;45671
                                ;45672          ;1-------------------------------;
                                ;45673          MTEMP0_MDR.OR.ZLIT8[20],        ; OR BIT 13 INTO LITERAL
U 04B9, 0127,2D92,4031,0047,004E,B  ;45674          CLEAR GFLOAT(FLAG4)             ;
                                ;45675
                                ;45676          ;--------------------------------;
                                ;45677          M[TEMP1]_R[TEMP0].ASL.1,        ; POSITION SIGN BIT CORRECTLY
U 04EB, 0086,15F7,0090,04E7,0000,1  ;45678          MDR_0,RETURN [+1]               ;
                                ;45679
                                ;45680 OS.DRED-IMMEDIATE:
                                ;45681          ;--------------------------------;
                                ;45682          MDR_XB PC_PC+1,                 ; GET SECOND HALF OF DATA
U 04EC, 0881,7002,403D,A467,0014,0  ;45683          NEXT/OS.WRT1                    ; GO DO A RETURN
```

```
                                ;45684 .TOC "  Native Mode Operand Specifier   : OS.DMOD"
                                ;45685
                                ;45686 ;*****************************************************************************
                                ;45687 ;       MDR is not saved and is destroyed in all but register mode.
                                ;45688 ;       Used to evaluate modify type operands for double data type
                                ;45689 ;       routines.  Bits 31-0 of the operand are left in TEMP1 and
                                ;45690 ;       bits 63-32 are left in MDR.  If the operand is in memory the
                                ;45691 ;       address of the lowest byte is left in VA.  If the operand is in
                                ;45692 ;       GPR's the register number of the lowest register is left in RNUM.
                                ;45693 ;       MTEMP0 is destroyed.
                                ;45694 ;       Write access to the operand is not checked.
                                ;45695 ;*****************************************************************************
                                ;45696
                                ;45697
                                ;45698
                                ;45699 190:
                                ;45700 OS.DMOD:
                                ;45701          ;0000-----------------------------; Rn          REGISTER MODE
                                ;45702          M[TEMP1] R[GPR.R],                ; TEMP0 HAS LOW HALF OF OPERAND
U 0190, 0086.15BE,403C,C047,004E,9 ;45703       NEXT/OS.DRED-REG                  ;
                                ;45704
                                ;45705 191:     ;0001-----------------------------; (Rn)+      AUTOINCREMENT MODE
                                ;45706          VA_R[GPR.R],                      ; ADDRESS OF LOW PART OF OPERAND
                                ;45707          CLOBBER MTEMP0,                   ; PUT GARBAGE IN MTEMP0
U 0191, 0C86,05BE,403C,C4A7,004E,D ;45708       NEXT/OS.DMOD-AUTO.INC             ;
                                ;45709
                                ;45710 192:     ;0010-----------------------------; I^#cons    IMMEDIATE MODE
U 0192, 0C80,0036,4030,0047,00FD,C ;45711       NEXT/IE.ADDR.MODE                 ; UNPREDICTABLE, TAKE ADDRESS MODE FAULT
                                ;45712
                                ;45713 193:     ;0011-----------------------------; S^#cons    LITERAL MODE
U 0193, 0C80,0036,4030,0047,00FD,C ;45714       NEXT/IE.ADDR.MODE                 ; ILLEGAL ADDRESSING MODE
                                ;45715
                                ;45716 194:     ;0100-----------------------------; -(Rn)      AUTODECREMENT MODE
                                ;45717          VA_R[GPR.R] RB-CONX.SIZ,          ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
U 0194, 0085,473C,003C,C4A7,0019,E ;45718       PUSH RBS-,NEXT/OS.DMOD-READ1      ; RBS GETS !RNUM!SIZE!-!
                                ;45719
                                ;45720 195:     ;0101-----------------------------; Addr       RELATIVE MODE
                                ;45721          VA_PC+SEXT(XB)+I PC_PC+I,         ; ADDRESS OF OPERAND IS PC + DSPLMT
                                ;45722          SIZE[IDEP],CLOBBER MTEMP0 DEF,    ; PUT GARBAGE IN MTEMP0
U 0195, 0487,7816,403D,8407,0019,E ;45723       NEXT/OS.DMOD-READ1                ;
                                ;45724
                                ;45725 196:     ;0110-----------------------------; D(Rn)      DISPLACEMENT MODE
                                ;45726          VA_SEXT(XB)+R[GPR.R] PC_PC+I,     ; ADDRESS OF OPERAND IS REG + DSPLMT
                                ;45727          SIZE[IDEP],CLOBBER MTEMP0 DEF,    ; PUT GARBAGE IN MTEMP0
U 0196, 0087,7815,003C,C4A7,0019,E ;45728       NEXT/OS.DMOD-READ1                ;
                                ;45729
                                ;45730 197:     ;0111-----------------------------; @#Addr     ABSOLUTE MODE
                                ;45731          VA_XB PC_PC+4,                    ; VA GETS ADDRESS OF OPERAND
                                ;45732          CLOBBER MTEMP0 DEF,               ; PUT GARBAGE IN MTEMP0
U 0197, 0C87,7002,402D,A4A7,0019,E ;45733       NEXT/OS.DMOD-READ1                ;
```

```
                                ;45734 198:    ;1000----------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;45735         VA_R[GPR.R],                     ; OPERAND ADDRESS IS GPR(RNUM)
                                ;45736         CLOBBER MTEMP0,                  ; PUT GARBAGE IN MTEMP0
U 0198, 0486,05BE,403C,C4A7,0019,E ;45737      NEXT/OS.DMOD-READ1               ;
                                ;45738
                                ;45739 199:    ;1001----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;45740         VA_PC+SEXT(XB)+I_PC_PC+I,        ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;45741         SIZE[IDEP],CLOBBER MTEMP0 DEF,   ; PUT GARBAGE IN MTEMP0
U 0199, 0C87,7816,403D,8407,004F,0 ;45742      NEXT/OS.DMOD-INDIRECT            ;
                                ;45743
                                ;45744 19A:    ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;45745         VA_SEXT(XB)+R[GPR.R]_PC_PC+I,    ; OPERAND IS REG + DATA FROM I-STREAM
                                ;45746         SIZE[IDEP],CLOBBER MTEMP0 DEF,   ; PUT GARBAGE IN MTEMP0
U 019A, 0887,7815,003C,C4A7,004F,0 ;45747      NEXT/OS.DMOD-INDIRECT            ;
                                ;45748
                                ;45749 19B:    ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;45750         VA_R[GPR.R],                     ; VA GETS OPERAND    GPR(RNUM)
                                ;45751         CLOBBER MTEMP0,                  ; PUT GARBAGE IN MTEMP0
U 019B, 0486,05BE,403C,C4A7,004F,2 ;45752      NEXT/OS.DMOD-READ.INC.4          ; SAVE MDR IN Q
                                ;45753
                                ;45754 19C:    ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 019C, 0C80,0036,4030,0047,00FD,C ;45755      NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45756
                                ;45757 19D:    ;1101----------------------------; (Rn)[Rx]  INDEX MODE
                                ;45758         M[TEMP0]_R[GPR.R],PUSH,          ; SAVE INDEX REGISTER IN TEMP0
U 019D, 0086,05BE,41FC,C047,0443,0 ;45759      BUT/LOD.BRA,NEXT/OS.BOA-QUAD     ; CALL ROUTINE TO GET BASE OPERAND ADD
                                ;45760
                                ;45761 19E:
                                ;45762 OS.DMOD-READ1:
                                ;45763         ;1110----------------------------;
                                ;45764         READ.MOD,SIZE[IDEP],             ; READ FIRST HALF OF OPERAND
                                ;45765         VA_VA+4,                         ; POINT VA TO NEXT HALF OF OPERAND
                                ;45766         CLOBBER MTEMP0 DEF,              ; PUT GARBAGE IN MTEMP0
U 019E, 0C86,0036,4030,0454,004E,E ;45767      NEXT/OS.DMOD-READ2               ;
                                ;45768
                                ;45769 OS.DMOD-AUTO.INC:
                                ;45770         ;--------------------------------;
                                ;45771         READ.MOD,                        ; READ FIRST HALF WITH MODIFY
                                ;45772         R[GPR.R]_RB+CONX.SIZ,            ; BUMP GPR
                                ;45773         PUSH RBS+,                       ; SAVE ORIGINAL STATE OF REGISTER
                                ;45774         VA_VA+4,                         ; POINT VA AT NEXT HALF
U 04ED, 0885,573D,003C,C454,004E,E ;45775      NEXT/OS.DMOD-READ2               ;
                                ;45776
                                ;45777 OS.DMOD-READ2:
                                ;45778         ;--------------------------------;
                                ;45779         READ.MOD,                        ; READ SECOND HALF OF OPERAND
                                ;45780         FPA_R[TEMP1]_M[MDR],SIZE[LONG],  ; SAVE MDR IN TEMP1
U 04EE, 0485,2592,4020,4054,104E,F ;45781      NEXT/OS.DMOD-DEC.VA              ;
                                ;45782
                                ;45783 OS.DMOD-DEC.VA:
                                ;45784         ;--------------------------------;
                                ;45785         VA_M[VA]-ZLIT0[4],               ; POINT VA BACK TO FIRST HALF
U 04EF, 0981,BC10,0070,24A7,0000,1 ;45786      IRDX [1]                         ;
```

```
                                ;45787 OS.DMOD-INDIRECT:
                                ;45788          ;--------------------------------;
                                ;45789          READ,SIZE[LONG],                ; GET ADDRESS OF ADDRESS OF OPERAND
U 04F0, 0080,0036,4020,0050,004F,1 ;45790       NEXT/OS.DMOD-VA_MDR             ;
                                ;45791
                                ;45792 OS.DMOD-VA_MDR:
                                ;45793          ;--------------------------------;
                                ;45794          VA_M[MDR],                      ; PUT ADDRESS IN VA
U 04F1, 0081,2002,403D,84A7,0019,E ;45795       NEXT/OS.DMOD-READ1
                                ;45796
                                ;45797 OS.DMOD-READ.INC.4:
                                ;45798          ;--------------------------------;
                                ;45799          READ,                           ; READ FIRST HALF OF OPERAND
                                ;45800          R[GPR.R]_RB+CONX(4),             ; BUMP GPR BY 4
                                ;45801          PUSH RBS+,                      ; SAVE STATE OF GPR
U 04F2, 0485,573D,002C,C050,004F,1 ;45802       NEXT/OS.DMOD-VA_MDR             ;
```

```
                                ;45803 .TOC "  Native Mode Operand Specifier   : OS.SOB"
                                ;45804
                                ;45805
                                ;45806 ;*************************************************************************
                                ;45807 ;       MDR is saved in the Q-register and is always destroyed.
                                ;45808 ;       Evaluates read type operands and places the result in MDR.
                                ;45809 ;       MTEMP0 is destroyed.
                                ;45810 ;       Read access for the operand is checked.
                                ;45811 ;       This routine is used only to speed up SOB register mode
                                ;45812 ;*************************************************************************
                                ;45813
                                ;45814 1A0:
                                ;45815 OS.SOB:
                                ;45816          ;0000-------------------------; Rn        REGISTER MODE
                                ;45817          Q_SEXT(XB) PC_PC+1,           ; MAKE SURE PC IS ACCESSABLE
U 01A0, 0881,7816,500D,A047,0076,7 ;45818       NEXT/CO.SOB-REG               ; GO TO CONTRL
                                ;45819
                                ;45820 1A1:     ;0001-------------------------; (Rn)+     AUTOINCREMENT MODE
                                ;45821          Q_M[MDR] VA_R[GPR.R],         ; PLACE ADDRESS OF OPERAND IN VA
                                ;45822          CLOBBER MTEMP0 DEF,           ; PUT GARBAGE IN MTEMP0
U 01A1, 0087,2004,703C,C4A7,004C,8 ;45823       NEXT/OS.MOD-AUTO.INC          ; INC GPR(RNUM) BY SIZE
                                ;45824
                                ;45825 1A2:     ;0010-------------------------; I^#cons   IMMEDIATE MODE
U 01A2, 0C80,0036,4030,0047,00FD,C ;45826       NEXT/IE.ADDR.MODE             ; UNPREDICTABLE, TAKE ADDRESS MODE FAULT
                                ;45827
                                ;45828 1A3:     ;0011-------------------------; S^#cons   LITERAL MODE
U 01A3, 0C80,0036,4030,0047,00FD,C ;45829       NEXT/IE.ADDR.MODE             ; ILLEGAL ADDRESSING MODE
                                ;45830
                                ;45831 1A4:     ;0100-------------------------; -(Rn)     AUTODECREMENT MODE
                                ;45832          VA_R[GPR.R]_RB-CONX.SIZ,      ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
                                ;45833          PUSH RBS-,                    ; VA GETS OPERAND ADDRESS
U 01A4, 0885,473C,003C,C4A7,004C,9 ;45834       NEXT/OS.MOD-READ.SAV.EXIT     ; RBS GETS !RNUM!SIZE!-!
                                ;45835
                                ;45836 1A5:     ;0101-------------------------; Addr      RELATIVE MODE
                                ;45837          VA_PC+SEXT(XB)+I PC_PC+I,     ; ADDRESS OF OPERAND IS PC + DSPLMT
                                ;45838          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 01A5, 0C87,7816,403D,8407,004C,9 ;45839       NEXT/OS.MOD-READ.SAV.EXIT     ;
                                ;45840
                                ;45841 1A6:     ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                                ;45842          VA_SEXT(XB)+R[GPR.R] PC_PC+I, ; ADDRESS OF OPERAND IS REG + DSPLMT
                                ;45843          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 01A6, 0887,7815,003C,C4A7,004C,9 ;45844       NEXT/OS.MOD-READ.SAV.EXIT     ; VA GETS ADDRESS OF OPERAND
                                ;45845
                                ;45846 1A7:     ;0111-------------------------; @#Addr    ABSOLUTE MODE
                                ;45847          VA_XB PC_PC+4,                ; NEXT 4 BYTES OF I-STREAM IS OPERAND
                                ;45848          CLOBBER MTEMP0 DEF,           ; PUT GARBAGE IN MTEMP0
U 01A7, 0487,7002,402D,A4A7,004C,9 ;45849       NEXT/OS.MOD-READ.SAV.EXIT     ; VA GETS ADDRESS OF OPERAND
                                ;45850
                                ;45851 1A8:     ;1000-------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;45852          Q_M[MDR] VA_R[GPR.R],         ; OPERAND ADDRESS IS GPR(RNUM)
                                ;45853          CLOBBER MTEMP0 DEF,           ; PUT GARBAGE IN MTEMP0
U 01A8, 0887,2004,703C,C4A7,0011,E ;45854       NEXT/OS.MOD-READ.EXIT         ;
```

```
                                ;45855 1A9:   ;1001----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;45856        VA_PC+SEXT(XB)+I PC_PC+I,       ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;45857        SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 01A9, 0C87,7816,403D,8407,004C,A ;45858     NEXT/OS.MOD-READ.SAV            ;
                                ;45859
                                ;45860 1AA:   ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;45861        VA_SEXT(XB)+R[GPR.R] PC_PC+I,   ; OPERAND IS REG + DATA FROM I-STREAM
                                ;45862        SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
U 01AA, 0887,7815,003C,C4A7,004C,A ;45863     NEXT/OS.MOD-READ.SAV            ;
                                ;45864
                                ;45865 1AB:   ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;45866        Q_M[MDR] VA_R[GPR.R],           ; VA GETS OPERAND    GPR(RNUM)
                                ;45867        CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
U 01AB, 0887,2004,703C,C4A7,004C,C ;45868     NEXT/OS.MOD-READ.INC.4          ; SAVE MDR IN Q
                                ;45869
                                ;45870 1AC:   ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 01AC, 0C80,0036,4030,0047,00FD,C ;45871     NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45872
                                ;45873 1AD:   ;1101----------------------------; (Rn)[Rx]  INDEX MODE
                                ;45874        MTEMP0_R[GPR.R] Q_MDR,          ; SAVE INDEX REGISTER IN TEMP0
U 01AD, 0487,2004,71FC,C047,0440,0 ;45875     PUSH,BOT/LOD.BRA,NEXT/OS.BOA    ; CALL ROUTINE TO GET BASE OPERAND ADD
                                ;45876
                                ;45877 1AE:
                                ;45878 OS.SOB-READ.EXIT:
                                ;45879        ;1110----------------------------;
                                ;45880        READ.MOD,SIZE[IDEP],            ; READ MEMORY AT VA  SIZE=DSIZE
U 01AE, 0080,0036,4070,0054,0000,1 ;45881     IRDX [1]                        ;
```

```
                                ;45882 .TOC "  Native Mode Operand Specifier   : OS.FIDRED"
                                ;45883
                                ;45884 ;*****************************************************************************
                                ;45885 ;       MDR is not saved and is destroyed in all but register
                                ;45886 ;       and immediate modes.
                                ;45887 ;       Evaluates read type operands for double data type
                                ;45888 ;       routines.  Bits 31-0 of the operand are left in TEMP1 and bits
                                ;45889 ;       63-32 are left in MDR.
                                ;45890 ;       MTEMP0 is destroyed.
                                ;45891 ;       Read access is checked.
                                ;45892 ;       This routine is used only by FPA interface.
                                ;45893 ;*****************************************************************************
                                ;45894
                                ;45895 1B0:
                                ;45896 OS.FIDRED:
                                ;45897          ;0000-------------------------; Rn        REGISTER MODE
                                ;45898          FPA_Q_MDR MTEMP0_R[GPR.R],     ; GET OPERAND FROM REGISTER
                                ;45899          SET FLAG0,                     ; SET FLAG FOR AUDREY/FPA
U 01B0, 0047,2004,703C,C047,104F,3 ;45900       NEXT/OS.FIDRED-REG             ;
                                ;45901
                                ;45902 1B1:     ;0001-------------------------; (Rn)+     AUTOINCREMENT MODE
                                ;45903          FPA_Q_M[MDR] VA_R[GPR.R],      ; ADDRESS OF LOW PART OF OPERAND
                                ;45904          CLEAR FLAG0,CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 01B1, 0007,2004,703C,C4A7,104E,0 ;45905       NEXT/OS.QRED-AUTO.INC          ;
                                ;45906
                                ;45907 1B2:     ;0C10-------------------------; I^#cons   IMMEDIATE MODE
                                ;45908          R[TEMP0]_XB PC_PC+1,           ; TEMP0 GETS FIRST HALF OF DATA
                                ;45909          SET FLAG0,                     ; SET FLAG FOR AUDREY/FPA
U 01B2, 0C45,7592,4030,2047,004F,4 ;45910       NEXT/OS.FIDRED.IMMED           ;
                                ;45911
                                ;45912 1B3:     ;0011-------------------------; S^#cons   LITERAL MODE
                                ;45913          FPA_Q_M[MDR].LITNXT,           ; PASS OPERAND TO FPA
                                ;45914          MDR_ZEXT(OSR),                 ; GET SHORT LITERAL
                                ;45915          SET FLAG1,                     ; SHORT LITERAL FLAG FOR POLYD
U 01B3, 0849,2592,5070,05E7,2000,1 ;45916       IRDX [1]                       ;
                                ;45917
                                ;45918 1B4:     ;0100-------------------------; -(Rn)     AUTODECREMENT MODE
                                ;45919          VA_R[GPR.R]_RB-CONX.SIZ,       ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
                                ;45920          CLEAR FLAG0,                   ; FOR AUDREY/FPA
                                ;45921          PUSH RBS-,                     ; RBS GETS !RNUM!SIZE!-!
U 01B4, 0805,473C,003C,C4A7,004E,3 ;45922       NEXT/OS.QRED-READ.SAV.INC      ;
                                ;45923
                                ;45924 1B5:     ;0101-------------------------; Addr      RELATIVE MODE
                                ;45925          VA_PC+SEXT(XB)+I PC_PC+I,      ; ADDRESS OF OPERAND IS PC + DSPLMT
                                ;45926          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
                                ;45927          CLEAR FLAG0,                   ; CLEAR FLAG FOR AUDREY/FPA
U 01B5, 0C07,7816,403D,8407,004E,3 ;45928       NEXT/OS.QRED-READ.SAV.INC      ;
                                ;45929
                                ;45930 1B6:     ;0110-------------------------; D(Rn)     DISPLACEMENT MODE
                                ;45931          VA_SEXT(XB)+RLGPR.R] PC_PC+I,  ; ADDRESS OF OPERAND IS REG + DSPLMT
                                ;45932          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
                                ;45933          CLEAR FLAG0,                   ; CLEAR FLAG FOR AUDREY/FPA
U 01B6, 0807,7815,003C,C4A7,004E,3 ;45934       NEXT/OS.QRED-READ.SAV.INC      ;
```

```
                                ;45935 1B7:   ;0111----------------------------; @#Addr    ABSOLUTE MODE
                                ;45936        VA_XB PC_PC+4,                  ; VA GETS ADDRESS OF OPERAND
                                ;45937        CLOBBER MTEMP0 DEF,             ; PUT GARBAGE IN MTEMP0
                                ;45938        CLEAR FLAG0,                    ; CLEAR FLAG FOR AUDREY
U 01B7, 0407,7002,402D,A4A7,004E,3 ;45939     NEXT/OS.QRED-READ.SAV.INC       ;
                                ;45940
                                ;45941 1B8:   ;1000----------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;45942        VA_R[GPR.R],CLEAR FLAG0,        ; OPERAND ADDRESS IS GPR(RNUM)
                                ;45943        CLOBBER MTEMP0,                 ; PUT GARBAGE IN MTEMP0
U 01B8, 0C06,05BE,403C,C4A7,004E,3 ;45944     NEXT/OS.QRED-READ.SAV.INC       ;
                                ;45945
                                ;45946 1B9:   ;1001----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;45947        VA_PC+SEXT(XB)+1 PC_PC+1,       ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;45948        SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
                                ;45949        CLEAR FLAG0,                    ; CLEAR FLAG FOR AUDREY
U 01B9, 0C07,7816,403D,8407,004E,5 ;45950     NEXT/OS.QRED-READ.IND           ;
                                ;45951
                                ;45952 1BA:   ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;45953        VA_SEXT(XB)+R[GPR.R] PC_PC+1,   ; OPERAND IS REG + DATA FROM I-STREAM
                                ;45954        SIZE[IDEP],CLOBBER MTEMP0 DEF,  ; PUT GARBAGE IN MTEMP0
                                ;45955        CLEAR FLAG0,                    ; CLEAR FLAG FOR AUDREY
U 01BA, 0807,7815,003C,C4A7,004E,5 ;45956     NEXT/OS.QRED-READ.IND           ;
                                ;45957
                                ;45958 1BB:   ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;45959        FPA_Q_M[MDR] VA_R[GPR.R],       ; VA GETS OPERAND    GPR(RNUM)
                                ;45960        CLEAR FLAG0,CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 01BB, 0807,2004,703C,C4A7,104E,7 ;45961     NEXT/OS.QRED-READ.INC.4         ; SAVE MDR IN Q
                                ;45962
                                ;45963 1BC:   ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 01BC, 0C80,0036,4030,0047,00FD,C ;45964     NEXT/IE.ADDR.MODE               ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;45965
                                ;45966 1BD:   ;1101----------------------------; (Rn)[Rx]  INDEX MODE
                                ;45967        FPA_Q_MDR MTEMP0_R[GPR.R],      ; GET INDEX REGISTER
                                ;45968        CLEAR FLAG0,PUSH,               ; CLEAR FLAG FOR AUDREY
U 01BD, 0407,2004,71FC,C047,1443,0 ;45969     BUT/LOD.BRA,NEXT/OS.BOA-QUAD    ; CALL ROUTINE TO GET BASE OPERAND ADD
                                ;45970
                                ;45971 1BE:   ;1110----------------------------;
                                ;45972        READ,SIZE[IDEP],                ; READ FIRST HALF OF OPERAND
                                ;45973        VA_VA+4,                        ; POINT VA TO NEXT HALF OF OPERAND
U 01BE, 0480,0036,4030,0450,004E,4 ;45974     NEXT/OS.QRED-READ2              ;
```

```
                                        ;45975 OS.FIDRED-REG:
                                        ;45976          ;-------------------------------;
                                        ;45977          FPA_M[TEMPO] MDR_R[GPR.R+1],    ; GET SECOND HALF OF OPERAND
04F3, 0080,05BE,407F,C467,1000,1        ;45978          IRDX [1]                        ;
                                        ;45979
                                        ;45980 OS.FIDRED.IMMED:
                                        ;45981          ;-------------------------------;
                                        ;45982          FPA_Q_M[MDR],                   ; PASS LAST OPERAND TO FPA
04F4, 0481,2592,5030,0047,104F,5        ;45983          NEXT/OS.FIDRED.READ2            ;
                                        ;45984
                                        ;45985 OS.FIDRED.READ2:
                                        ;45986          ;-------------------------------;
04F5, 0481,7002,402D,A467,004F,6        ;45987          MDR_XB PC_PC+4                  ; GET SECOND HALF OF OPERAND
                                        ;45988
                                        ;45989          ;-------------------------------;
                                        ;45990          FPA_M[TEMPO],                   ; PASS OPERAND TO FPA
04F6, 0080,0036,4070,0047,1000,1        ;45991          IRDX [1]                        ;
```

```
                        ;45992 .TOC "  Native Mode Operand Specifier   : OS.FIDMOD"
                        ;45993
                        ;45994 ;*****************************************************************************
                        ;45995 ;       MDR is not saved and is destroyed in all but register
                        ;45996 ;       and immediate modes.
                        ;45997 ;       Evaluates read type operands for double data type
                        ;45998 ;       routines.  Bits 31-0 of the operand are left in TEMP1 and bits
                        ;45999 ;       63-32 are left in MDR.
                        ;46000 ;       MTEMP0 is destroyed.
                        ;46001 ;       Write access is checked.
                        ;46002 ;       This routine is used only by FPA interface.
                        ;46003 ;*****************************************************************************
                        ;46004
                        ;46005 1C0:
                        ;46006 OS.FIDMOD:
                        ;46007          ;0000-------------------------; Rn        REGISTER MODE
                        ;46008          FPA_Q_MDR MTEMP0_R[GPR.R],     ; GET OPERAND FROM REGISTER
U 01C0, 0087,2004,703C,C047,104F,3      ;46009          NEXT/OS.FIDRED-REG             ;
                        ;46010
                        ;46011 1C1:     ;0001-------------------------; (Rn)+      AUTOINCREMENT MODE
                        ;46012          FPA_Q_M[MDR] VA_R[GPR.R],      ; ADDRESS OF LOW PART OF OPERAND
                        ;46013          CLOBBER MTEMP0 DEF,            ; PUT GARBAGE IN MTEMP0
U 01C1, 0087,2004,703C,C4A7,104E,D      ;46014          NEXT/OS.DMOD-AUTO.INC          ;
                        ;46015
                        ;46016 1C2:     ;0010-------------------------; I^#cons    IMMEDIATE MODE
U 01C2, 0C80,0036,4030,0047,00FD,C      ;46017          NEXT/IE.ADDR.MODE              ; UNPREDICTABLE, TAKE ADDRESS MODE FAULT
                        ;46018
                        ;46019 1C3:     ;0011-------------------------; S^#cons    LITERAL MODE
U 01C3, 0C80,0036,4030,0047,00FD,C      ;46020          NEXT/IE.ADDR.MODE              ; ILLEGAL ADDRESSING MODE
                        ;46021
                        ;46022 1C4:     ;0100-------------------------; -(Rn)      AUTODECREMENT MODE
                        ;46023          VA_R[GPR.R]_RB-CONX.SIZ,       ; OPERAND ADDRESS IS GPR(RNUM) - SIZE
                        ;46024          PUSH RBS-,SIZE[IDEP],          ; RBS GETS |RNUM|SIZE|-|
U 01C4, 0085,473C,003C,C4A7,004F,7      ;46025          NEXT/OS.FIDMOD-READ1           ;
                        ;46026
                        ;46027 1C5:     ;0101-------------------------; Addr       RELATIVE MODE
                        ;46028          VA_PC+SEXT(XB)+1_PC_PC+1,      ; ADDRESS OF OPERAND IS PC + DSPLMT
                        ;46029          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 01C5, 0487,7816,403D,8407,004F,7      ;46030          NEXT/OS.FIDMOD-READ1           ;
                        ;46031
                        ;46032 1C6:     ;0110-------------------------; D(Rn)      DISPLACEMENT MODE
                        ;46033          VA_SEXT(XB)+R[GPR.R]_PC_PC+1,  ; ADDRESS OF OPERAND IS REG + DSPLMT
                        ;46034          SIZE[IDEP],CLOBBER MTEMP0 DEF, ; PUT GARBAGE IN MTEMP0
U 01C6, 0087,7815,003C,C4A7,004F,7      ;46035          NEXT/OS.FIDMOD-READ1           ;
                        ;46036
                        ;46037 1C7:     ;0111-------------------------; @#Addr     ABSOLUTE MODE
                        ;46038          VA_XB_PC_PC+4,                 ; VA GETS ADDRESS OF OPERAND
                        ;46039          CLOBBER MTEMP0 DEF,            ; PUT GARBAGE IN MTEMP0
U 01C7, 0C87,7002,402D,A4A7,004F,7      ;46040          NEXT/OS.FIDMOD-READ1           ;
```

```
                                ;46041 1C8:    ;1000----------------------------; (Rn)      REGISTER DEFERRED MODE
                                ;46042         VA_R[GPR.R],                     ; OPERAND ADDRESS IS GPR(RNUM)
                                ;46043         CLOBBER MTEMP0,                  ; PUT GARBAGE IN MTEMP0
U 01C8, 0486,058E,403C,C4A7,004F,7 ;46044      NEXT/OS.FIDMOD-READ1             ;
                                ;46045
                                ;46046 1C9:    ;1001----------------------------; @Addr     RELATIVE DEFERRED MODE
                                ;46047         VA_PC+SEXT(XB)+I PC_PC+I,        ; VA GETS ADDRESS OF ADDRESS OF OPERAND
                                ;46048         SIZE[IDEP],CLOBBER MTEMP0 DEF,   ; PUT GARBAGE IN MTEMP0
U 01C9, 0487,7816,403D,8407,004F,8 ;46049      NEXT/OS.FIDMOD-INDIRECT          ;
                                ;46050
                                ;46051 1CA:    ;1010----------------------------; @D(Rn)    DISPLACEMENT DEFERRED MODE
                                ;46052         VA_SEXT(XB)+R[GPR.R] PC_PC+I,    ; OPERAND IS REG + DATA FROM I-STREAM
                                ;46053         SIZE[IDEP],CLOBBER MTEMP0 DEF,   ; PUT GARBAGE IN MTEMP0
U 01CA, 0087,7815,003C,C4A7,004F,8 ;46054      NEXT/OS.FIDMOD-INDIRECT          ;
                                ;46055
                                ;46056 1CB:    ;1011----------------------------; @(Rn)+    AUTO-INCREMENT DEFERRED MODE
                                ;46057         FPA_Q_M[MDR] VA_R[GPR.R],        ; VA GETS OPERAND    GPR(RNUM)
                                ;46058         CLOBBER MTEMP0 DEF,              ; PUT GARBAGE IN MTEMP0
U 01CB, 0887,2004,703C,C4A7,104F,2 ;46059      NEXT/OS.DMOD-READ.INC.4          ; SAVE MDR IN Q
                                ;46060
                                ;46061 1CC:    ;1100----------------------------; (Rn)[PC]  INDEX MODE PC
U 01CC, 0C80,0036,4030,0047,00FD,C ;46062      NEXT/IE.ADDR.MODE                ; ILLEGAL ADDRESSING MODE  TELL CHARLIE
                                ;46063
                                ;46064 1CD:    ;1101----------------------------; (Rn)[Rx]  INDEX MODE
                                ;46065         FPA_Q_MDR MTEMP0_R[GPR.R],PUSH,  ; GET INDEX REGISTER
U 01CD, 0C87,2004,71FC,C047,1443,0 ;46066      BUT7LOD.BRA,NEXT7OS.BOA-QUAD     ; CALL ROUTINE TO GET BASE OPERAND ADD
                                ;46067
                                ;46068 1CE:    ;1110----------------------------;
                                ;46069         READ.MOD,SIZE[IDEP],             ; READ FIRST HALF OF OPERAND
                                ;46070         VA_VA+4,                         ; POINT VA TO NEXT HALF OF OPERAND
U 01CE, 0C80,0036,4030,0454,004E,E ;46071      NEXT/OS.DMOD-READ2               ;
                                ;46072
                                ;46073 OS.FIDMOD-READ1:
                                ;46074         ;-------------------------------;
                                ;46075         READ.MOD,SIZE[IDEP],             ; GET FIRST HALF OF OPERAND
                                ;46076         FPA_Q_M[MDR],                    ; SAVE MDR
                                ;46077         CLOBBER MTEMP0 DEF,              ; GARBAGE MTEMP0
                                ;46078         VA_VA+4,                         ; POINT TO NEXT HALF
U 04F7, 0487,2592,5030,0454,104E,E ;46079      NEXT/OS.DMOD-READ2               ; FINNISH IN REGULAR DMOD
                                ;46080
                                ;46081 OS.FIDMOD-INDIRECT:
                                ;46082         ;-------------------------------;
                                ;46083         READ.SIZE[LONG],                 ; GET ADDRESS OF OPERAND
                                ;46084         FPA_Q_M[MDR],                    ; SAVE MDR
U 04F8, 0881,2592,5020,0050,104F,1 ;46085      NEXT/OS.DMOD-VA_MDR              ; FINNISH IN REGULAR DMOD
```

```
                                        ;46086 .TOC "UVERFY.MIC"
                                        ;46087 .TOC "REVISION 10.0"
                                        ;46088 ;       CHARLIE MCDOWELL
                                        ;46089
;46090  .NOBIN
;46091
;46092  .TOC "  Revision History"
;46093
;46094  ; REV   EXPLAINATION
;46095
;46096  ; 10    Clean up DSIZE test.
;46097  ; 09    Clean up DSIZE test.
;46098  ; 08    Remove free locations 173A and 173B
;46099  ; 07    Initial release.
                                        ;46100          .BIN
```

```
;46101 .TOC "  MICRO VERIFY                        : TEST INITIALIZATION"
;46102
;46103 ;*************************************************************************
;46104 ;       The micro verify routines are invoked by doing a PUSH,NEXT/MV.TEST and
;46105 ;       upon successful completion a RETURN [+1] is done.
;46106 ;
;46107 ;       At the start of micro verify, a '%' is typed at the console.  If
;46108 ;       all tests run successfully, a second '%' is typed.  If an error is
;46109 ;       detected, a single error character (see list below) is typed at the
;46110 ;       console and then the CPU is halted in console mode.  When the CPU
;46111 ;       halts an error code will be typed in place of the PC (see list below)
;46112 ;       and a halt code of /tbs/ which will indicate this halt is due to an
;46113 ;       error condition in micro verify.
;46114 ;
;46115 ;       The following is a list or the test sections and their associated
;46116 ;       error codes (for a complete error list see CHARTS):
;46117 ;
;46118 ;               PC+2(see below) CHAR            TEST NAME
;46119 ;               0000000X        @               R-BUS, W-BUS, D-REG TEST
;46120 ;               0000003X        C               M-BUS, Q-REG TEST
;46121 ;               0000005X        E               SCRATCH PAD TEST
;46122 ;               0000006X        F               SCRATCH PAD EXPLICIT ADDR MTEMPS
;46123 ;               0000009X        I               SCRATCH PAD EXPLICIT ADDR RTEMPS
;46124 ;               000000AX        J               SCRATCH PAD EXPLICIT ADDR IPRS
;46125 ;               000000CX        L               SCRATCH PAD EXPLICIT ADDR GPRS
;46126 ;               000000CE        L               SCRATCH PAD DUAL PORT ADDRESS
;46127 ;               000000FX        O               XB, IR, OSR TEST
;46128 ;               0000011X        Q               XB, PC_PC+ISIZE TEST
;46129 ;               0000012X        R               DSIZE TEST
;46130 ;               0000014X        T               DSIZE TEST
;46131 ;               0000018X        X               CACHE PARITY CHECKER
;46132 ;               000001BX        [               TB PARITY CHECKER
;46133 ;               000001DX        ]               CONTROL STORE PARITY CHECKER
;46134 ;               000001EX        ^               CACHE TEST
;46135 ; *****NOTE*****
;46136 ;       The PC that is typed out on the console is 2 less than the value that
;46137 ;       micro verify loads into the PC.  To find the exact point of the error
;46138 ;       you should first add 2 to the PC then look in the listing for
;46139 ;       PC_ZLIT0[nnn] where nnn is the value of the typed out PC + 2.
;46140 ;*************************************************************************
```

```
                                ;46141 .REGION/MV.R1L,MV.R1H/MV.R2L,MV.R2H/MV.R3L,MV.R3H
                                ;46142 162C:    ;*********FORCE ADDRESS*********;
                                ;46143 MV.TEST:
                                ;46144          ;0------------------------------;
U 162C, 0C80,0036,4030,0047,0487,9  ;46145      PUSH,NEXT/IN.INIT               ; DO INIT BEFORE TESTING
                                ;46146
                                ;46147 162D:    ;1------------------------------;
U 162D, 01A0,0CB7,0030,7687,0163,0  ;46148      MEMSCAR_ZLIT24[0E],STEPC_2      ; LOAD ADDRESS OF CMI DISABLE REGISTER
                                ;46149
                                ;46150 =00      ;00-----------------------------;
                                ;46151          M[TEMP5]_ZLIT0[0D],DEC STEPC,   ; TYPE <CR><LF>
U 1630, 059E,5C37,0030,6847,049F,4  ;46152      PUSH,NEXT/CN.TYPE.CHAR          ; STEPC MUST BE 1. (RETURNED AS 1 ALSO)
                                ;46153
                                ;46154          ;01-----------------------------;
                                ;46155          M[TEMP5]_ZLIT0[25],             ; TYPE % TO INDICATE START OF TEST.
U 1631, 0586,5C37,0031,2847,049F,4  ;46156      PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;46157
                                ;46158          ;10-----------------------------;
U 1632, 0C80,0E77,0030,0607,0179,5  ;46159      MEMSCR_-1                       ; DISABLE THE CMI
                                ;46160 =
                                ;46161          ;-------------------------------;
U 1795, 0D81,DC37,2030,7847,0179,7  ;46162      D_RNUM_ZLIT0[0F]                ; PREPARE TO SAVE GPRS
                                ;46163
                                ;46164          ;-------------------------------;
                                ;46165          VA_ZLIT0[13C],                  ;
U 1797, 0D80,0C37,0039,E4A7,0163,9  ;46166      NEXT/MV.SAVE.GPRS.01            ; SAVE THEM IN CACHE 100-13C
                                ;46167 =00
                                ;46168 MV.SAVE.GPRS:
                                ;46169          ;00-----------------------------;
                                ;46170          WRITE R[GPR.R],SIZE[LONG],      ;
U 1638, 0080,05BE,402C,C5D8,0179,C  ;46171      NEXT/MV.FINISH.SETUP            ;
                                ;46172
                                ;46173 MV.SAVE.GPRS.01:
                                ;46174          ;01-----------------------------;
                                ;46175          WRITE R[GPR.R],SIZE[LONG],      ;
U 1639, 0880,05BE,402C,C5D8,057D,D  ;46176      PUSH,NEXT/MV.STALL.FOR.CACHE    ;
                                ;46177
                                ;46178          ;10-----------------------------;
U 163A, 0581,BC10,0030,24A7,0179,9  ;46179      VA_M[VA]-ZLIT0[4]               ;
                                ;46180 =
                                ;46181          ;-------------------------------;
                                ;46182          D_RNUM_D-ZLIT0[1],              ;
U 1799, 0581,DC30,2A70,0847,0963,8 378*  ;46183  WX.NE.0?,NEXT/MV.SAVE.GPRS      ;
                                ;46184
                                ;46185 MV.FINISH.SETUP:
                                ;46186          ;-------------------------------;
U 179C, 0484,05B7,0032,44E7,0179,D  ;46187      R[TEMP9]_0,MDR_0                ; MDR_0 TO ALLOW CACHE WRITE TO FINISH.
```

```
                                ;46188 .TOC "  MICRO VERIFY                    : R-BUS, W-BUS, D-REG TEST"
                                ;46189
                                ;46190 ;*************************************************************************
                                ;46191 ;TEST R-BUS, W-BUS, AND D-REG
                                ;46192 ;       RTMP8 HOLDS THE ALP TEST VALUE
                                ;46193 ;       MTMP8 HOLDS THE BASE VALUE USED TO GENERATE SUPROT TEST VALUES
                                ;46194 ;       MTMP9 HOLDS THE SHIFT COUNT USED TO GENERATE SUPROT TEST VALUES
                                ;46195 ;               SUPROT TEST VALUE = MTMP8 ROTATED LEFT MTMP9 BITS
                                ;46196 ;
                                ;46197 ;*************************************************************************
                                ;46198
                                ;46199 MV.BUS.TEST:
                                ;46200         ;-------------------------------; LOAD THE BASE ERROR CODE FOR
U 179D, 0D86,BDB7,0030,3047,0179,E ;46201       M[ERRCOD]_ZLIT8[6]              ; BUS TEST ERRORS (6XX)
                                ;46202
                                ;46203         ;***************************************
                                ;46204         ;VERIFY WITH A 1 IN A FIELD OF 0'S
                                ;46205         ;***************************************
                                ;46206
                                ;46207         ; DO INITIALIZATION OF GENERATING A 1 IN BIT0 FROM
                                ;46208         ; THE ALP AND THE SUPROT, ALSO INITIALIZE THE LOOP COUNT IN
                                ;46209         ; THE STEP COUNTER SHOULD ALREADY BE ZERO.
                                ;46210
                                ;46211         ;-------------------------------;
U 179E, 0086,9BFE,303D,8047,0179,F ;46212       PL_M[TEMP9]_D_Q_0               ;
                                ;46213
                                ;46214         ;-------------------------------; GET ALP TEST VALUE
U 179F, 0484,00A9,2032,0047,017A,0 ;46215       R[TEMP8]_D_D+Q+1                ; DREG=00000001         RTMP8=00000001
                                ;46216
                                ;46217         ;-------------------------------; GET SUPROT BASE TEST VALUE
                                ;46218         M[TEMP8]_ZLIT0[1],              ; MTMP8=00000001
U 17A0, 0586,8C37,0030,0847,017A,2 ;46219       NEXT/MV.RWBUS.FLOAT1.10         ; SKIP 1ST PART OF LOOP THE 1ST TIME
                                ;46220
                                ;46221         ;****BEGIN LOOP LP1*****; WALK A 1 IN A FIELD OF 0'S FROM BIT0 TO BIT32
                                ;46222
                                ;46223 =0
                                ;46224 MV.RWBUS.FLOAT1:
                                ;46225         ;0-----------------------------; GET  ALP TEST VALUE
                                ;46226         R[TEMP8]_D_(D+RB).SL.1 Q<0>_1,  ; DREG=...010...        RTMP8=...010...
U 1648, 0884,0425,C032,0047,017A,1 ;46227       NEXT/MV.RWBUS.FLOAT1.5          ;
                                ;46228
                                ;46229         ;1-----------------------------; END OF RWBUS.FLOAT1 LOOP
                                ;46230         PL_M[TEMP9]_D_Q_0,              ; QREG=DREG=00000000    MTMP9=00000000
U 1649, 0086,9BFE,303D,8047,017A,3 ;46231       NEXT/MV.MBUS.TEST               ;
                                ;46232
                                ;46233 MV.RWBUS.FLOAT1.5:
                                ;46234         ;-------------------------------;  ADD 1 TO THE SHIFT COUNT
U 17A1, 0C86,98C9,0030,0047,017A,2 ;46235       PL_M[TEMP9]_MB+Q                ; MTMP9=MTMP9+1          PLATCH=MTMP9
```

```
                                        ;46236 MV.RWBUS.FLOAT1.10:
                                        ;46237          ;-------------------------------;
                                        ;46238          Q_D.XOR.(M[TEMP8].RL.P),        ; COMPARE ALP AND SUPROT TEST VALUES
U 17A2, 0C80,8873,5A30,0047,0964,C 325* ;46239          WX.EQ.0?                        ;
                                        ;46240 =0
                                        ;46241 MV.ERR.605:
                                        ;46242          ;0-------------------------------; BAD D-REG OR SUPROT
U 164C, 0D80,0C37,0030,0487,003E,F      ;46243          PC_ZLIT0[000],NEXT/MV.ERROR     ;
                                        ;46244
                                        ;46245          ;1-------------------------------; COMPARE OK
                                        ;46246          D_Q_D.XOR.R[TEMP8],             ;TEST R&WBUS(RTMP8 WAS WRITTEN VIA WBUS)
U 164D, 0C80,0023,7A32,0047,0165,C      ;46247          WX.EQ.0?                        ; QREG=DREG=00000000
                                        ;46248 =0
                                        ;46249 MV.ERR.606:
                                        ;46250          ;0-------------------------------; BAD RBUS OR WBUS
U 165C, 0980,0C37,0030,0C87,003E,F      ;46251          PC_ZLIT0[001],NEXT/MV.ERROR     ;
                                        ;46252
                                        ;46253          ;1-------------------------------; COMPARE OK    STEPCNTR=STEPCNTR-1
U 165D, 0C80,0036,4330,0047,0164,8      ;46254          DBZ STEPC?,NEXT/MV.RWBUS.FLOAT1 ;
                                        ;46255
                                        ;46256          ;*****END LOOP LP1******;
```

```
                                ;46257 .TOC "  MICRO VERIFY                          : M-BUS Q-REG TEST"
                                ;46258          ;*************************************************************************
                                ;46259          ;TEST M-BUS AND Q-REG
                                ;46260          ;       MTMP8 HOLDS THE ALP TEST VALUE
                                ;46261          ;       MTMP9 HOLDS THE SUPROT TEST VALUE SHIFT COUNT
                                ;46262          ;       MTMP0 HOLDS THE SUPROT BASE TEST VALUE
                                ;46263          ;               SUPROT TEST VALUE = MTMP0 ROTATED LEFT MTMP9 BITS
                                ;46264          ;
                                ;46265          ;*************************************************************************
                                ;46266
                                ;46267          ;****************************************
                                ;46268          ;VERIFY WITH A 1 IN A FIELD OF 0'S
                                ;46269          ;****************************************
                                ;46270          ; DO INITIALIZATION OF GENERATING A 1 IN BIT0 FROM
                                ;46271          ; THE ALP AND THE SUPROT, ALSO INITIALIZE THE LOOP COUNT IN
                                ;46272          ; THE STEP COUNTER.
                                ;46273
                                ;46274 MV.MBUS.TEST:
                                ;46275          ;-------------------------------; GET ALP TEST VALUE
U 17A3, 0C86,80A9,3030,0047,017A,4      ;46276          M[TEMP8]_D_Q_D+Q+1              ; DREG=00000001         MTMP8=00000001
                                ;46277
                                ;46278          ;-------------------------------; GET SUPROT BASE TEST VALUE
                                ;46279          M[TEMP0]_ZLIT0[1],              ; MTMP0=00000001
U 17A4, 0986,0C37,0030,0847,017A,6      ;46280          NEXT/MV.MBUS.FLOAT1.10          ;
                                ;46281
                                ;46282          ;****BEGIN LOOP LP3*****; WALK A 1 IN A FIELD OF 0'S FROM BIT0 TO BIT32
                                ;46283 =0
                                ;46284 MV.MBUS.FLOAT1:
                                ;46285          ;0------------------------------; GET ALP TEST VALUE
                                ;46286          M[TEMP8]_D_(MB+Q).SL.1 Q<0>_1,  ; DREG=...010...        MTMP8=...010...
U 166C, 0086,840D,E030,0047,017A,5      ;46287          NEXT/MV.MBUS.FLOAT1.05          ;
                                ;46288
                                ;46289          ;1------------------------------; END MBUS TEST, BEGIN SP ADDRESS TEST
                                ;46290          LONLIT_[11111111],              ; INCREMENT FOR ALL 8 NIBBLES
U 166D, 0780,0777,7777,7047,017A,8      ;46291          NEXT/MV.SP.TEST                 ;
                                ;46292
                                ;46293 MV.MBUS.FLOAT1.05:
                                ;46294          ;-------------------------------;  ADD 1 TO THE SHIFT COUNT
U 17A5, 0084,0BF9,0032,4047,017A,6      ;46295          PL_R[TEMP9]_RB+Q                ; PLATCH=RTMP9=RTMP9+1
                                ;46296
                                ;46297 MV.MBUS.FLOAT1.10:
                                ;46298          ;-------------------------------; GET SUPROT TEST VALUE
U 17A6, 0880,0877,1030,0047,017A,7      ;46299          Q_M[TEMP0].RL.P                 ; QREG=...010...
```

```
                                        ;46300          ;--------------------------------; TEST Q-REG
U 17A7, 0080,002B,6A30,0047,0168,C      ;46301          D_D.XOR.Q,WX.EQ.0?               ; DREG=00000000
                                        ;46302 =0
                                        ;46303 MV.ERR.60D:
                                        ;46304          ;0--------------------------------; BAD Q-REG
U 168C, 0980,0C37,0031,8C87,003E,F      ;46305          PC_ZLIT0[031],NEXT/MV.ERROR      ;
                                        ;46306
                                        ;46307          ;1--------------------------------; COMPARE OK
                                        ;46308          D_Q_M[TEMP8].XOR.Q,              ; TEST MBUS
U 168D, 0C80,800B,7A30,0047,016A,C      ;46309          WX.EQ.0?                         ; QREG=DREG=00000000
                                        ;46310 =0
                                        ;46311 MV.ERR.60E:
                                        ;46312          ;0--------------------------------; BAD MBUS
U 16AC, 0D80,0C37,0031,9C87,003E,F      ;46313          PC_ZLIT0[033],NEXT/MV.ERROR      ;
                                        ;46314
                                        ;46315          ;1--------------------------------; COMPARE OK
U 16AD, 0C80,0036,4330,0047,0166,C      ;46316          DBZ STEPC?,NEXT/MV.MBUS.FLOAT1   ;
                                        ;46317          ;*****END LOOP LP3******         ;
```

```
                        ;46318 .TOC "  MICRO VERIFY                    : SCRATCH PAD TEST"
                        ;46319
                        ;46320 ;*****************************************************************************
                        ;46321 ;
                        ;46322 ;       Resources       DREG            Hold register number for loading RNUM.
                        ;46323 ;                       RNUM            Used for addressing scratch pads.
                        ;46324 ;                       STEPC           Used to loop on filling s.p. with ones.
                        ;46325 ;                       QREG            Temp used to fill s.p. with ones.
                        ;46326 ;                       FLAG1           Set to flag end of s.p. test.
                        ;46327 ;
                        ;46328 ;       Output          PC              Error code if error exit.
                        ;46329 ;
                        ;46330 ;       This section does a memory test of the 48 scratch pads that are
                        ;46331 ;       driven onto the R-bus (16 RTEMPs, 16 GPRs, and 16 IPRs) and the
                        ;46332 ;       16 scratch pads that are driven onto the M-bus (the 8 dual ports are
                        ;46333 ;       tested twice).
                        ;46334 ;       First all 56 (not 64 since 8 are dual port) scratch pads are zeroed.
                        ;46335 ;       Then starting with RTEMP0 to RTEMP15(MM.TEMP1) each scratch pad is
                        ;46336 ;       checked for zero and then filled with ones, one bit at a time.
                        ;46337 ;       This is repeated for GPR0 to GPR15(RTEMPGPR), IPR0(KSP) to
                        ;46338 ;       IPR15(MM.TEMP4) and MTEMP8 to MTEMP15(SISR).
                        ;46339 ;
                        ;46340 ;*****************************************************************************
                        ;46341
                        ;46342 MV.SP.TEST:
                        ;46343          ;-------------------------------; LOAD RNUM TO ZERO ALL 16 SCRATCH
U 17A8, 0581,DC37,2030,7847,0161,5      ;46344          D_RNUM_ZLIT0[0F]                ; PADS IN EACH OF THE 4 GROUPS
                        ;46345 =01
                        ;46346 MV.SP.ZERO:
                        ;46347          ;01-------------------------------;
                        ;46348          R[TEMP.R]_0,MDR_0,              ; ZERO RTEMP0-RTEMP15(MM.TEMP1)
U 1615, 0C84,05B7,003C,04E7,017A,9      ;46349          NEXT/MV.SP.ZERO.GPR             ;
                        ;46350
                        ;46351          ;11-------------------------------;
                        ;46352          STEPC_D_RNUM_ZLIT0[0],          ; INIT RNUM FOR START OF TEST
U 1617, 0981,DC37,2030,0107,0158,F      ;46353          NEXT/MV.SP.TEST.RTEMPS          ;
                        ;46354
                        ;46355 MV.SP.ZERO.GPR:
                        ;46356          ;-------------------------------;
U 17A9, 0484,05B7,003C,C047,017A,A      ;46357          R[GPR.R]_0                      ; ZERO GPR0-GPR15(RTMPGPR)
                        ;46358
                        ;46359          ;-------------------------------;
U 17AA, 0884,05B7,003C,8047,017A,B      ;46360          R[IPR.R]_0                      ; ZERO IPR0(KSP)-IPR15(MM.TEMP4)
                        ;46361
                        ;46362          ;-------------------------------;
                        ;46363          CLEAR FLAG1,                    ; CLEAR TO INDICATE BEGINNING OF TEST
U 17AB, 0J0F,05BE,403D,8047,017A,C      ;46364          M[TEMP.R]_R[ZERO]               ; ZERO MTEMP0-MTEMP15(SISR)
                        ;46365
                        ;46366          ;-------------------------------;
                        ;46367          D_RNUM_D-ZLIT0[1],SIZE[LONG],   ; LOOP TO ZERO ALL 16 TEMPS
U 17AC, 0581,DC30,2B60,0847,0961,5 413* ;46368          SIGND CMP DEF?,NEXT/MV.SP.ZERO  ;
```

```
                                ;46369 =001111
                                ;46370 MV.SP.TEST.RTEMPS:
                                ;46371          ;001111-------------------------;
                                ;46372          Q_R[TEMP.R],WX.EQ.0?,           ; CHECK IF RTEMP IS ZERO
U 158F, 0080,05BE,5A3C,0047,096B,8 322*  ;46373          NEXT/MV.SP.RTEMP.ADR.ERR        ;
                                ;46374
                                ;46375 MV.SP.TEST.GPRS:
                                ;46376          ;011111-------------------------;
                                ;46377          Q_R[GPR.R],WX.EQ.0?,            ; CHECK IF GPR IS ZERO
U 159F, 0880,05BE,5A3C,C047,096C,2 322*  ;46378          NEXT/MV.SP.GPR.ADR.ERR          ;
                                ;46379
                                ;46380 MV.SP.TEST.IPRS:
                                ;46381          ;101111-------------------------;
                                ;46382          Q_R[IPR.R],WX.EQ.0?,            ; CHECK IF IPR IS ZERO
U 15AF, 0C80,05BE,5A3C,8047,096C,8 322*  ;46383          NEXT/MV.SP.IPR.ADR.ERR          ;
                                ;46384
                                ;46385          ;111111-------------------------;
                                ;46386          D_RNUM_D+ZLIT0[8],              ; SKIP DUAL PORTS
U 15BF, 0581,DC31,2030,4047,015F,F       ;46387          NEXT/MV.SP.TEST.MTEMPS          ;
                                ;46388 =0
                                ;46389 MV.SP.RTEMP.ADR.ERR:
                                ;46390          ;0------------------------------; RTEMP WAS NOT ALL ZERO
U 16B8, 0980,0C37,0032,8C87,003E,F       ;46391          PC_ZLIT0[051],NEXT/MV.ERROR     ; SET ERROR CODE.
                                ;46392
                                ;46393          ;1------------------------------;
U 16B9, 0084,0737,100C,0047,016B,A       ;46394          R[TEMP.R]_Q_CONX(1)             ; PREPARE TO FILL RTEMP WITH ONES
                                ;46395 =0
                                ;46396 MV.SP.CHECK.RTEMPS:
                                ;46397          ;0------------------------------;
                                ;46398          WB_R[TEMP.R].XOR.Q,WX.EQ.0?,    ; COMPARE RTEMP AND Q
U 16BA, 0880,003B,4A3C,0047,016C,0       ;46399          NEXT/MV.SP.RTEMP.ERROR          ;
                                ;46400
                                ;46401          ;1------------------------------;
                                ;46402          D_RNUM_D+ZLIT0[1],WB<5-0>?,     ; INCREMENT RTEMP ADDRESS AND CHECK
U 16BB, 0181,DC31,2230,0847,0958,F 377*  ;46403          NEXT/MV.SP.TEST.RTEMPS          ; IF DONE WITH RTEMPS
                                ;46404 =0
                                ;46405 MV.SP.RTEMP.ERROR:
                                ;46406          ;0------------------------------; ERROR FILLING RTEMP WITH ONES,
U 16C0, 0980,0C37,0032,9487,003E,F       ;46407          PC_ZLIT0[052],NEXT/MV.ERROR     ; SET ERROR CODE AND EXIT.
                                ;46408
                                ;46409          ;1------------------------------;
                                ;46410          R[TEMP.R]_Q_((M[MDR]+Q).SL.1).OR.1, ; SET ANOTHER BIT IN RTEMP
                                ;46411          DBZ STEPC?,                     ; LOOP UNTIL ALL ONES
U 16C1, 0485,2109,D33C,0047,016B,A       ;46412          NEXT/MV.SP.CHECK.RTEMPS         ;
                                ;46413 =0
                                ;46414 MV.SP.GPR.ADR.ERR:
                                ;46415          ;0------------------------------; GPR NOT ALL ZEROS,
U 16C2, 0580,0C37,0032,A487,003E,F       ;46416          PC_ZLIT0[054],NEXT/MV.ERROR     ; SET ERROR CODE AND EXIT.
                                ;46417
                                ;46418          ;1------------------------------;
U 16C3, 0084,0737,100C,C047,016C,4       ;46419          R[GPR.R]_Q_CONX(1)              ; PREPARE TO FILL GPR WITH ONES
```

```
                                        ;46420 =0
                                        ;46421 MV.SP.CHECK.GPRS:
                                        ;46422          ;0-------------------------------;
                                        ;46423          WB_R[GPR.R].XOR.Q,WX.EQ.0?,     ; COMPARE GPR AND Q
U 16C4, 0880,003B,4A3C,C047,016C,6      ;46424          NEXT/MV.SP.GPR.ERROR            ;
                                        ;46425
                                        ;46426          ;1-------------------------------;
                                        ;46427          D_RNUM_D+ZLIT0[1],WB<5-0>?,     ; INCREMENT GPR ADDRESS AND CHECK
U 16C5, 0181,DC31,2230,0847,0958,F 377* ;46428          NEXT/MV.SP.TEST.RTEMPS          ; IF DONE WITH GPRS
                                        ;46429 =0
                                        ;46430 MV.SP.GPR.ERROR:
                                        ;46431          ;0-------------------------------; ERROR FILLING GPR WITH ONES,
U 16C6, 0580,0C37,0032,BC87,003E,F      ;46432          PC_ZLIT0[057],NEXT/MV.ERROR     ; SET ERROR CODE AND EXIT.
                                        ;46433
                                        ;46434          ;1-------------------------------;
                                        ;46435          R[GPR.R]_Q_((M[MDR]+Q).SL.1).OR.1, ; SET ANOTHER BIT IN GPR
                                        ;46436          DBZ STEP0?,                     ; LOOP UNTIL ALL ONES
U 16C7, 0485,2109,D33C,C047,016C,4      ;46437          NEXT/MV.SP.CHECK.GPRS           ;
                                        ;46438 =0
                                        ;46439 MV.SP.IPR.ADR.ERR:
                                        ;46440          ;0-------------------------------; IPR NOT ALL ZERO,
U 16C8, 0980,0C37,0032,C487,003E,F      ;46441          PC_ZLIT0[058],NEXT/MV.ERROR     ; SET ERROR CODE AND EXIT.
                                        ;46442
                                        ;46443          ;1-------------------------------;
U 16C9, 0C84,0737,100C,8047,016C,A      ;46444          R[IPR.R]_Q_CONX(1)              ; PREPARE TO FILL IPR WITH ONES
                                        ;46445 =0
                                        ;46446 MV.SP.CHECK.IPRS:
                                        ;46447          ;0-------------------------------;
                                        ;46448          WB_R[IPR.R].XOR.Q,WX.EQ.0?,     ; COMPARE IPR AND Q
U 16CA, 0480,003B,4A3C,8047,016D,0      ;46449          NEXT/MV.SP.IPR.ERROR            ;
                                        ;46450
                                        ;46451          ;1-------------------------------;
                                        ;46452          D_RNUM_D+ZLIT0[1],WB<5-0>?,     ; INCREMENT IPR ADDRESS AND CHECK
U 16CB, 0181,DC31,2230,0847,0958,F 377* ;46453          NEXT/MV.SP.TEST.RTEMPS          ; IF DONE WITH IPRS
                                        ;46454 =0
                                        ;46455 MV.SP.IPR.ERROR:
                                        ;46456          ;0-------------------------------; ERROR FILLING IPR WITH ONES,
U 16D0, 0980,0C37,0032,DC87,003E,F      ;46457          PC_ZLIT0[05B],NEXT/MV.ERROR     ; SET ERROR CODE AND EXIT.
                                        ;46458
                                        ;46459          ;1-------------------------------;
                                        ;46460          R[IPR.R]_Q_((M[MDR]+0: SL.1).OR.1, ; SET ANOTHER BIT IN IPR
                                        ;46461          DBZ STEP0?,                     ; LOOP UNTIL ALL ONES
U 16D1, 0885,2109,D33C,8047,016C,A      ;46462          NEXT/MV.SP.CHECK.IPRS           ;
```

```
                                ;46463 =011111
                                ;46464 MV.SP.MTEMP.LOOP:
                                ;46465          ;011111-------------------------; BEGIN SP EXPLICIT ADDRESS TEST.
                                ;46466          R[MM.TEMP1]_D_Q_-1,             ; TEST VALUE(8 NIBBLES OF ADDRESS)
                                ;46467          CLEAR FLAGO,                    ;
U 15DF, 0004,0E77,3033,C047,017A,D ;46468       NEXT/MV.SP.EXPLICIT.ADR         ;
                                ;46469
                                ;46470 MV.SP.TEST.MTEMPS:
                                ;46471          ;111111-------------------------;
U 15FF, 0081,0592,5A30,0047,096D,2 322* ;46472  Q_M[TEMP.R],WX.EQ.0?             ; CHECK IF MTEMP IS ZEPO
                                ;46473
                                ;46474 =0       ;0-------------------------------; MTEMP IS NOT ALL ZERO,
U 16D2, 0580,0C37,0032,EC87,003E,F ;46475       PC_ZLIT0[05D],NEXT/MV.ERROR     ; SET ERROR CODE AND EXIT.
                                ;46476
                                ;46477          ;1-------------------------------;
U 16D3, 0187,0C37,1030,0847,016D,4 ;46478       M[TEMP.R]_Q_ZLIT0[1]            ; PREPARE TO FILL M; WITH ONES
                                ;46479 =0
                                ;46480 MV.SP.CHECK.MTEMPS:
                                ;46481          ;0-------------------------------;
                                ;46482          WB_M[TEMP.R].XOR.Q Q_(Q.SL.1).OR.1, ; SHIFT IN ANOTHER ONE
                                ;46483          WX.EQ.0?,                       ; COMPARE MTEMP AND Q
U 16D4, 0481,010F,4A30,0047,016D,6 ;46484       NEXT/MV.SP.MTEMP.ERROR          ;
                                ;46485
                                ;46486          ;1-------------------------------;
                                ;46487          D_RNUM_D+ZLIT0[1],WB<5-0>?,     ; INCREMENT MTEMP ADDRESS AND CHECK
U 16D5, 0181,0C31,2230,0847,095D,F 377* ;46488  NEXT/MV.SP.MTEMP.LOOP            ; IF DONE WITH MTEMPS
                                ;46489 =0
                                ;46490 MV.SP.MTEMP.ERROR:
                                ;46491          ;0-------------------------------; ERROR FILLING MTEMP WITH ONES,
U 16D6, 0580,0C37,0032,F487,003E,F ;46492       PC_ZLIT0[05E],NEXT/MV.ERROR     ; SET ERROR CODE AND EXIT.
                                ;46493
                                ;46494          ;1-------------------------------;
                                ;46495          M[TEMP.R] Q,DBZ STEPC?,         ; SET ANOTHER BIT IN MTEMP
U 16D7, 0887,003A,435D,3047,016D,4 ;46496       NEXT/MV.SP.CHECK.MTEMPS         ;
```

```
;46497 .TOC "  MICRO VERIFY                    : SCRATCH PAD EXPLICIT ADDRESS TEST"
;46498
;46499 ;*************************************************************************
;46500 ;
;46501 ;       Resources       LONLIT          Constant 11111111
;46502 ;                       QREG    Temp to hold test value.
;46503 ;                       DREG            Hold register number for loading RNUM.
;46504 ;                       RNUM            Used for addressing scratch pads.
;46505 ;                       MTEMP8          Constant 11111111 for R-bus scratch pads
;46506 ;                                          {  0 - loading RTEMPs.
;46507 ;                       FLAG<1:0>          {  1 - loading GPRs.
;46508 ;                                          {  2 - not used.
;46509 ;                                          {  3 - loading MTEMP8 (end setup).
;46510 ;
;46511 ;       This section tests that the scratch pads can be addressed explicitly
;46512 ;       instead of using RNUM (ie MSRC/TEMP0 instead of MSRC/TEMP.R).
;46513 ;       First all R-bus scratch pads and MTEMP8 are loaded with their
;46514 ;       address in all 8 nibbles(ie RTEMP3=33333333, GPR14=EEEEEEEE).
;46515 ;       Then the registers with addresses 0,1,2,4 and 8 are read explicitly.
;46516 ;       The dual ports are written only from RSRC but are read from both sides.
;46517 ;*************************************************************************
;46518
;46519 MV.SP.EXPLICIT.ADR:
;46520          ;-------------------------------;
;46521          M[TEMP8]_R[LONLIT],             ; LOAD DECREMENT FOR RTEMP TESTS
;46522          CLEAR FLAG1,                    ;
U 17AD, 080E,85BE,403D,4047,016D,8 ;46523          NEXT/MV.SP.ADR.LP               ;
;46524
;46525 =00
;46526 MV.SP.RTEMP.ADR:
;46527          ;00-------------------------------;
;46528          R[TEMP.R]_Q Q-M[TEMP8],         ; WRITE ADDRESS OF RTEMP TO ALL 8 NIBS
U 1640, 0084,800B,1A3C,0047,096D,8 344* ;46529          WX.EQ.0?,NEXT/MV.SP.ADR.LP      ;
;46530
;46531 MV.SP.GPR.ADR:
;46532          ;01-------------------------------;
;46533          R[GPR.R]_Q Q-M[TEMP8],          ; WRITE ADDRESS OF GPR TO ALL 8 NIBS
U 1641, 0084,800B,1A3C,C047,096D,8 344* ;46534          WX.EQ.0?,NEXT/MV.SP.ADR.LP      ;
;46535 =11
;46536 MV.SP.IPR.ADR:
;46537          ;11-------------------------------;
;46538          R[IPR.R]_Q Q-M[TEMP8],          ; WRITE ADDRESS OF IPR TO ALL 8 NIBS
U 1643, 0484,800B,1A3C,8047,096D,8 344* ;46539          WX.EQ.0?,NEXT/MV.SP.ADR.LP      ;
```

```
                                ;46540 =0
                                ;46541 MV.SP.ADR.LP:
                                ;46542          ;0------------------------------;
                                ;46543          D_RNUM_D-ZLIT0[1],FLAG<1-0>?,   ; LOOP DECREMENTING RNUM TO WRITE
U 16D8, 0981,DC30,2530,0847,0164,0 ;46544       NEXT/MV.SP.RTEMP.ADR            ; ALL 16 RETMPS/GPRS/IPRS
                                ;46545
                                ;46546          ;1------------------------------; USE FLAGS TO MOVE FROM RTEMPS TO GPRS
U 16D9, 0C80,0036,4530,0047,0164,4 ;46547       FLAG<1-0>?                      ; TO IPRS THEN MTEMP8
                                ;46548 =00
                                ;46549          ;00-----------------------------;
                                ;46550          R[RTMPGPR]_D_Q_RNUM_-1,         ; TEST VALUE (8 NIBBLES OF ADDRESS)
                                ;46551          SET FLAG0,                      ; SET FLAG TO TEST GPRS
U 1644, 0C45,DE77,3037,C047,016D,8 ;46552       NEXT/MV.SP.ADR.LP               ; LOOP BACK TO LOAD GPRS
                                ;46553
                                ;46554          ;01-----------------------------;
                                ;46555          R[MM.TEMP4]_D_Q_RNUM_-1,        ; TEST VALUE (8 NIBBLES OF ADDRESS)
                                ;46556          SET FLAG1,                      ; SET FLAG TO TEST IPRS
U 1645, 044D,DE77,303B,C047,016D,8 ;46557       NEXT/MV.SP.ADR.LP               ;
                                ;46558
                                ;46559 =11      ;11-----------------------------;
U 1647, 0C86,85BE,4032,0047,017A,E ;46560       M[TEMP8]_R[TEMP8]               ; PUT 88888888 IN MTEMP8
                                ;46561
                                ;46562          ;------------------------------;
U 17AE, 0080,05BE,503D,4047,017A,F ;46563       Q_R[LONLIT]                     ; PUT CHECK VALUE IN DREG=11111111
                                ;46564
                                ;46565          ;------------------------------;
U 17AF, 0880,0592,4A30,0047,096D,A 322* ;46566  WB_M[TEMP0],WX.EQ.0?             ; MTEMP0 SHOULD BE 0
                                ;46567
                                ;46568 =0       ;0------------------------------;
U 16DA, 0980,0C37,0033,0C87,003E,F ;46569       PC_ZLIT0[061],NEXT/MV.ERROR     ;
                                ;46570
                                ;46571          ;1------------------------------;
                                ;46572          WB_M[TEMP1].XOR.Q Q_Q.RL.1,     ; MTEMP1 SHOULD BE 11111111
U 16DB, 0080,130F,4A30,0047,016E,0 ;46573       WX.EQ.0?                        ; SHIFT FOR NEXT CHECK VALUE
                                ;46574
                                ;46575 =0       ;0------------------------------;
U 16E0, 0980,0C37,0033,1487,003E,F ;46576       PC_ZLIT0[062],NEXT/MV.ERROR     ;
                                ;46577
                                ;46578          ;1------------------------------;
                                ;46579          WB_M[TEMP2].XOR.Q Q_Q.RL.1,     ; MTEMP2 SHOULD BE 22222222
U 16E1, 0880,230F,4A30,0047,016E,2 ;46580       WX.EQ.0?                        ; SHIFT FOR NEXT CHECK VALUE
                                ;46581
                                ;46582 =0       ;0------------------------------;
U 16E2, 0580,0C37,0033,2487,003E,F ;46583       PC_ZLIT0[064],NEXT/MV.ERROR     ;
                                ;46584
                                ;46585          ;1------------------------------;
                                ;46586          WB_M[TEMP4].XOR.Q Q_Q.RL.1,     ; MTEMP4 SHOULD BE 44444444
U 16E3, 0880,430F,4A30,0047,016E,4 ;46587       WX.EQ.0?                        ; SHIFT FOR NEXT CHECK VALUE
                                ;46588
                                ;46589 =0       ;0------------------------------;
U 16E4, 0580,0C37,0033,3C87,003E,F ;46590       PC_ZLIT0[067],NEXT/MV.ERROR     ;
                                ;46591
                                ;46592          ;1------------------------------;
U 16E5, 0480,800B,4A30,0047,016E,6 ;46593       WB_M[TEMP8].XOR.Q,WX.EQ.0?      ; MTEMP8 SHOULD BE 88888888
```

```
                                ;46594 =0       ;0------------------------------;
U 16E6, 0980,0C37,0033,4487,003E,F  ;46595      PC_ZLIT0[068],NEXT/MV.ERROR     ;
                                ;46596
                                ;46597          ;1------------------------------;
U 16E7, 0480,05BE,4A30,0047,096E,8 322*  ;46598 WB_R[TEMP0],WX.EQ.0?             ; RTEMP0 SHOULD BE 0
                                ;46599
                                ;46600 =0       ;0------------------------------;
U 16E8, 0980,0C37,0034,8C87,003E,F  ;46601      PC_ZLIT0[091],NEXT/MV.ERROR     ;
                                ;46602
                                ;46603          ;1------------------------------;
                                ;46604          WB_M[TEMP1].XOR.R[TEMP1],       ; RTEMP1 SHOULD BE 11111111
U 16E9, 0480,1003,4A30,4047,016E,A  ;46605      WX.EQ.0?                        ;
                                ;46606
                                ;46607 =0       ;0------------------------------;
U 16EA, 0980,0C37,0034,9487,003E,F  ;46608      PC_ZLIT0[092],NEXT/MV.ERROR     ;
                                ;46609
                                ;46610          ;1------------------------------;
                                ;46611          WB_M[TEMP2].XOR.R[TEMP2],       ; RTEMP2 SHOULD BE 22222222
U 16EB, 0480,2003,4A30,8047,016E,C  ;46612      WX.EQ.0?                        ;
                                ;46613
                                ;46614 =0       ;0------------------------------;
U 16EC, 0580,0C37,0034,A487,003E,F  ;46615      PC_ZLIT0[094],NEXT/MV.ERROR     ;
                                ;46616
                                ;46617          ;1------------------------------;
                                ;46618          WB_M[TEMP4].XOR.R[TEMP4],       ; RTEMP4 SHOULD BE 44444444
U 16ED, 0C80,4003,4A31,0047,016F,0  ;46619      WX.EQ.0?                        ;
                                ;46620
                                ;46621 =0       ;0------------------------------;
U 16F0, 0580,0C37,0034,BC87,003E,F  ;46622      PC_ZLIT0[097],NEXT/MV.ERROR     ;
                                ;46623
                                ;46624          ;1------------------------------;
                                ;46625          WB_M[TEMP8].XOR.R[TEMP8],       ; RTEMP8 SHOULD BE 88888888
U 16F1, 0480,8003,4A32,0047,016F,2  ;46626      WX.EQ.0?                        ;
                                ;46627
                                ;46628 =0       ;0------------------------------;
U 16F2, 0980,0C37,0034,C487,003E,F  ;46629      PC_ZLIT0[098],NEXT/MV.ERROR     ;
                                ;46630
                                ;46631          ;1------------------------------;
U 16F3, 0880,05BE,4A38,0047,096F,4 322*  ;46632 WB_R[KSP],WX.EQ.0?              ; IPR0 (KSP) SHOULD BE 0
                                ;46633
                                ;46634 =0       ;0------------------------------;
U 16F4, 0980,0C37,0035,0C87,003E,F  ;46635      PC_ZLIT0[0A1],NEXT/MV.ERROR     ;
                                ;46636
                                ;46637          ;1------------------------------;
                                ;46638          WB_M[TEMP1].XOR.R[ESP],         ; IPR1 (ESP) SHOULD BE 11111111
U 16F5, 0880,1003,4A38,4047,016F,6  ;46639      WX.EQ.0?                        ;
                                ;46640
                                ;46641 =0       ;0------------------------------;
U 16F6, 0980,0C37,0035,1487,003E,F  ;46642      PC_ZLIT0[0A2],NEXT/MV.ERROR     ;
                                ;46643
                                ;46644          ;1------------------------------;
                                ;46645          WB_M[TEMP2].XOR.R[SSP],         ; IPR2 (SSP) SHOULD BE 22222222
U 16F7, 0080,2003,4A38,8047,016F,8  ;46646      WX.EQ.0?                        ;
```

```
                                ;46647 =0       ;0------------------------------;
U 16F8, 0580,0C37,0035,2487,003E,F      ;46648          PC_ZLIT0[0A4],NEXT/MV.ERROR     ;
                                ;46649
                                ;46650          ;1------------------------------;
                                ;46651          WB_M[TEMP4].XOR.R[ISP],         ; IPR4 (ISP) SHOULD BE 44444444
U 16F9, 0880,4003,4A39,0047,016F,A      ;46652          WX.EQ.0?                        ;
                                ;46653
                                ;46654 =0       ;0------------------------------;
U 16FA, 0580,0C37,0035,3C87,003E,F      ;46655          PC_ZLIT0[0A7],NEXT/MV.ERROR     ;
                                ;46656
                                ;46657          ;1------------------------------;
                                ;46658          WB_M[TEMP8].XOR.R[P0BR],        ; IPR8 (P0BR) SHOULD BE 88888888
U 16FB, 0880,8003,4A3A,0047,016F,C      ;46659          WX.EQ.0?                        ;
                                ;46660
                                ;46661 =0       ;0------------------------------;
U 16FC, 0980,0C37,0035,4487,003E,F      ;46662          PC_ZLIT0[0A8],NEXT/MV.ERROR     ;
                                ;46663
                                ;46664          ;1------------------------------;
U 16FD, 0880,05BE,4A34,0047,096F,E 322* ;46665          WB_R[R0],WX.EQ.0?               ; R0 SHOULD BE 0
                                ;46666
                                ;46667 =0       ;0------------------------------;
U 16FE, 0980,0C37,0036,0C87,003E,F      ;46668          PC_ZLIT0[0C1],NEXT/MV.ERROR     ;
                                ;46669
                                ;46670          ;1------------------------------;
                                ;46671          WB_M[TEMP1].XOR.R[R1],          ; R1 SHOULD BE 11111111
U 16FF, 0080,1003,4A34,4047,0170,0      ;46672          WX.EQ.0?                        ;
                                ;46673
                                ;46674 =0       ;0------------------------------;
U 1700, 0980,0C37,0036,1487,003E,F      ;46675          PC_ZLIT0[0C2],NEXT/MV.ERROR     ;
                                ;46676
                                ;46677          ;1------------------------------;
                                ;46678          WB_M[TEMP2].XOR.R[R2],          ; R2 SHOULD BE 22222222
U 1701, 0880,2003,4A34,8047,0170,2      ;46679          WX.EQ.0?                        ;
                                ;46680
                                ;46681 =0       ;0------------------------------;
U 1702, 0580,0C37,0036,2487,003E,F      ;46682          PC_ZLIT0[0C4],NEXT/MV.ERROR     ;
                                ;46683
                                ;46684          ;1------------------------------;
                                ;46685          WB_M[TEMP4].XOR.R[R4],          ; R4 SHOULD BE 44444444
U 1703, 0880,4003,4A35,0047,0170,4      ;46686          WX.EQ.0?                        ;
                                ;46687
                                ;46688 =0       ;0------------------------------;
U 1704, 0580,0C37,0036,3C87,003E,F      ;46689          PC_ZLIT0[0C7],NEXT/MV.ERROR     ;
                                ;46690
                                ;46691          ;1------------------------------;
                                ;46692          WB_M[TEMP8].XOR.R[R8],          ; R8 SHOULD BE 88888888
U 1705, 0080,8003,4A36,0047,0170,6      ;46693          WX.EQ.0?                        ;
                                ;46694
                                ;46695 =0       ;0------------------------------;
U 1706, 0980,0C37,0036,4487,003E,F      ;46696          PC_ZLIT0[0C8],NEXT/MV.ERROR     ;
```

```
                                ;46697 .TOC "  MICRO VERIFY                    : SCRATCH PAD DUAL PORT ADDRESS TEST"
                                ;46698
                                ;46699 ;*************************************************************************
                                ;46700 ;
                                ;46701 ;       Resources       DREG            Hold register number for loading RNUM.
                                ;46702 ;                       RNUM            Used for addressing TEMPs in a loop.
                                ;46703 ;
                                ;46704 ;       Output          PC              Error code if error exit.
                                ;46705 ;
                                ;46706 ;       This section checks that if the dual ports are written from the
                                ;46707 ;       'M' side then they can be read from the 'R' side.  Other tests have
                                ;46708 ;       done partial implicit testing of writing from the 'R' side and
                                ;46709 ;       reading from the 'M' side.
                                ;46710 ;       The 8 dual ports are written with 0-7 using SPW/MLONG and
                                ;46711 ;       MSRC/TEMP.R.  They are then read and checked using RSRC/TEMP.R.
                                ;46712 ;*************************************************************************
                                ;46713
                                ;46714 MV.SP.DUAL.PORT:
                                ;46715         ;1-----------------------------;
U 1707, 0181,DC37,2030,3847,0165,1  ;46716         D_RNUM_ZLIT0[07]              ;
                                ;46717 =01
                                ;46718 MV.SP.DUAL.PORT.WRT:
                                ;46719         ;01----------------------------;
                                ;46720         M[TEMP.R]_D,                  ; LOAD REGISTER WITH ADDRESS
U 1651, 0887,05B2,4030,0047,017B,0  ;46721         NEXT/MV.SP.DUAL.PORT.LP       ;
                                ;46722
                                ;46723         ;11----------------------------;
                                ;46724         D_RNUM_ZLIT0[7],              ; LOOP OVER 8 DUAL PORTS
U 1653, 0981,DC37,2030,3847,0165,5  ;46725         NEXT/MV.SP.DUAL.PORT.READ     ;
                                ;46726
                                ;46727 MV.SP.DUAL.PORT.LP:
                                ;46728         ;----------------------------;
                                ;46729         D_RNUM_D-ZLIT0[1],SIZE[LONG],
                                ;46730         SIGND CMP DEF?,               ; DECREMENT REGISTER ADDRESS AND LOOP
U 17B0, 0581,DC30,2B60,0847,0965,1 413*  ;46731         NEXT/MV.SP.DUAL.PORT.WRT      ;
                                ;46732 =01
                                ;46733 MV.SP.DUAL.PORT.READ:
                                ;46734         ;01----------------------------;
                                ;46735         WB_D.XOR.R[TEMP.R],WX.EQ.0?,  ; READ DUAL PORT ON R-BUS AND CHECK
U 1655, 0880,0023,4A3C,0047,0170,8  ;46736         NEXT/MV.SP.DUAL.PORT CHK      ;
                                ;46737
                                ;46738         ;11----------------------------;
                                ;46739         D_RNUM_ZLIT0[0],              ; PREPARE TO RESTORE GPRS.
U 1657, 0981,DC37,2030,0047,017B,1  ;46740         NEXT/MV.RESTORE.GPRS          ;
                                ;46741 =0
                                ;46742 MV.SP.DUAL.PORT.CHK:
                                ;46743         ;0-----------------------------;
U 1708, 0580,0C37,0036,7487,003E,F  ;46744         PC_ZLIT0[0CE],NEXT/MV.ERROR   ;
                                ;46745
                                ;46746         ;1-----------------------------;
                                ;46747         D_RNUM_D-ZLIT0[1],            ;
                                ;46748         SIGND CMP DEF?,SIZE[LONG],    ; DECREMENT REGISTER ADDRESS AND LOOP
U 1709, 0D81,DC30,2B60,0847,0965,5 413*  ;46749         NEXT/MV.SP.DUAL.PORT.READ     ;
```

```
                                        ;46750 .TOC "  MICRO VERIFY                    : RESTORE GPRS"
                                        ;46751
                                        ;46752 ;*************************************************************************
                                        ;46753 ;
                                        ;46754 ;       INPUT           VA MUST BE 100(HEX)
                                        ;46755 ;
                                        ;46756 ;       This section restores GPR1-5 and RTMPGPR which all contain values
                                        ;46757 ;       used by the boot and/or console code.
                                        ;46758 ;       They had been previously saved in CACHE locations 100-114(hex).
                                        ;46759 ;*************************************************************************
                                        ;46760
                                        ;46761 MV.RESTORE.GPRS:
                                        ;46762          ;-------------------------------;
U 17B1, 0980,0C37,0030,8107,017B,2      ;46763          STEPC_ZLITO[10]                 ; RESTORE ALL 16 GPRS.
                                        ;46764
                                        ;46765 MV.RESTORE.GPR.10:
                                        ;46766          ;-------------------------------;
                                        ;46767          READ,SIZE[LONG],VA_VA+4,        ;
U 17B2, 0880,0036,4320,0450,0170,A      ;46768          DBZ STEPC?                      ;
                                        ;46769
                                        ;46770 =0       ;0-----------------------------;
                                        ;46771          R[GPR.R]_M[MDR],                ;
U 170A, 0C85,2592,403C,C047,017B,3      ;46772          NEXT/MV.RESTORE.GPR.LP          ;
                                        ;46773
                                        ;46774          ;1-----------------------------;
                                        ;46775          R[RTMPGPR]_M[MDR],              ;
U 170B, 0085,2592,4037,C047,017B,4      ;46776          NEXT/MV.XB.TEST                 ; GO TO NEXT TEST.
                                        ;46777
                                        ;46778 MV.RESTORE.GPR.LP:
                                        ;46779          ;-------------------------------;
                                        ;46780          D_RNUM_D+ZLITO[1],              ;
U 17B3, 0D81,DC31,2030,0847,017B,2      ;46781          NEXT/MV.RESTORE.GPR.10          ;
```

```
                                ;46782 .TOC "  MICRO VERIFY                    : XB, IR, AND OSR TEST"
                                ;46783
                                ;46784 ;*****************************************************************************
                                ;46785 ;
                                ;46786 ;       Resources       STEPC           Loop counter for floating 1.
                                ;46787 ;                       TEMP0           Test pattern for XB<31:0>.
                                ;46788 ;                       TEMP1           Test pattern for XB<63:32>.
                                ;46789 ;                       VA              Used to write test pattern to cache.
                                ;46790 ;                       MICRO-STACK     Clobbered by MV.CHECK.XB if error exit.
                                ;46791 ;
                                ;46792 ;       Output          PC              Error code if error exit.
                                ;46793 ;
                                ;46794 ;       Subroutines     MV.CHECK.XB
                                ;46795 ;
                                ;46796 ;       This section floats a 1 in a field of 0's through all 64 bits of the XB
                                ;46797 ;       and the IR and OSR. The IR and OSR are tested with the low 2 bytes
                                ;46798 ;       of the XB each time the XB is tested.  This results in the IR and
                                ;46799 ;       OSR being tested redundantly with many 0's but uses very little code.
                                ;46800 ;       The test pattern (1 in a field of 0's) is floated through TEMP0
                                ;46801 ;       and TEMP1 each time calling MV.CHECK.XB which write TEMP0 and TEMP1
                                ;46802 ;       to the cache then source the data through the XB to check it.
                                ;46803 ;*****************************************************************************
                                ;46804
                                ;46805 MV.XB.TEST:
                                ;46806         ;-------------------------------;
U 17B4, 0186,1C37,0030,0047,017B,5  ;46807         M[TEMP1]_ZLIT0[0]               ; SETUP TO LOOP 32 TIMES, XB<63:32>=0
                                ;46808
                                ;46809         ;-------------------------------;
U 17B5, 0986,0C37,0030,0847,0165,8  ;46810         M[TEMP0]_ZLIT0[1]               ; SETUP TO FLOAT A 1 THRU XB<31:0>
                                ;46811 =00
                                ;46812 MV.XB.TEST.LOOP1:
                                ;46813         ;00-------------------------------;
                                ;46814         VA_ZLIT0[0],                    ; TEST PATTERN IS WRITTEN AT 0.
U 1658, 0580,0C37,0030,04A7,0570,C  ;46815         PUSH,NEXT/MV.CHECK.XB           ;
                                ;46816
                                ;46817         ;01-------------------------------;
                                ;46818         M[TEMP0]_ZLIT0[0],      ; SETUP TO LOOP 32 TIMES, XB<31:0>=0
U 1659, 0586,0C37,0030,0047,017B,6  ;46819         NEXT/MV.XB.TEST.HIGH.BITS       ;
                                ;46820
                                ;46821         ;10-------------------------------;
                                ;46822         M[TEMP0]_MB.RL.1,WB<0>?,        ; FLOAT A 1 THRU XB<31:0>
U 165A, 0C86,0301,C2BD,8047,0965,8 355*  ;46823         NEXT/MV.XB.TEST.LOOP1           ;
                                ;46824 =
```

```
                                        ;46825 MV.XB.TEST.HIGH.BITS:
                                        ;46826          ;-------------------------------;
U 17B6, 0586,1C37,0030,0847,0166,0      ;46827          M[TEMP1]_ZLIT0[1]               ; SETUP TO FLOAT A 1 THRU XB<63:32>
                                        ;46828 =00
                                        ;46829 MV.XB.TEST.LOOP2:
                                        ;46830          ;00-------------------------------;
                                        ;46831          VA_ZLIT0[0],                    ; TEST PATTERN IS WRITTEN AT 0.
U 1660, 0580,0C37,0030,04A7,0570,C      ;46832          PUSH,NEXT/MV.CHECK.XB           ;
                                        ;46833
                                        ;46834          ;01-------------------------------;
U 1661, 0C80,0036,4030,0047,017B,C      ;46835          NEXT/MV.XB.PC.TEST              ;
                                        ;46836
                                        ;46837          ;10-------------------------------;
                                        ;46838          M[TEMP1]_MB.RL.1,WB<0>?,        ; FLOAT A 1 THRU XB<63:32>
U 1662, 0086,1301,C2BD,8047,0966,0 355* ;46839          NEXT/MV.XB.TEST.LOOP2           ;
                                        ;46840 =
```

```
                    ;46841 .TOC "  MICRO VERIFY                    :       MV.CHECK.XB"
                    ;46842
                    ;46843 ;*************************************************************************
                    ;46844 ;
                    ;46845 ;       Inputs          TEMP0           Test pattern for XB<31:0>.
                    ;46846 ;                       TEMP1           Test pattern for XB<63:32>.
                    ;46847 ;                       VA              Must be zero.
                    ;46848 ;                       ****THE CMI SHOULD BE DISABLED OR MEMORY WILL BE WRITTEN
                    ;46849 ;
                    ;46850 ;       Resources       MDR             Clobbered as side effect.
                    ;46851 ;                       PC              Used to source XB and to load IR and OSR
                    ;46852 ;                       TEMP7           Used to hold TEMP0<15:8> for testing OSR
                    ;46853 ;
                    ;46854 ;       Output          PC              Error code if error exit.
                    ;46855 ;                       MICRO-STACK     Clobbered if error exit.
                    ;46856 ;                       VA              Result of compare if error exit.
                    ;46857 ;
                    ;46858 ;       This subroutine writes TEMP0 and TEMP1 into the first 64 bits of the
                    ;46859 ;       cache, then points PC to 0 to fill the XB and compares the XB with
                    ;46860 ;       TEMP0 and TEMP1.  It then points PC to 0 and uses IRD1TEST to load
                    ;46861 ;       IR and OSR and compares them with TEMP0<7:0> and TEMP0<15:8>.
                    ;46862 ;       In order for main memory to not be changed, the calling routine must
                    ;46863 ;       make sure that the CMI is disabled.
                    ;46864 ;
                    ;46865 ;*************************************************************************
                    ;46866 =0
                    ;46867 MV.CHECK.XB:
                    ;46868          ;0-------------------------------;
                    ;46869          R[TEMP7]_M[TEMP0],              ; SAVE INPUT IN A TEMP
                    ;46870          WRITE,SIZE[LONG],               ; WRITE PATTERN FOR XB<31:0> TO CACHE
U 170C, 0C84,0592,4021,C5D8,057D,D  ;46871          PUSH,NEXT/MV.STALL.FOR.CACHE    ;
                    ;46872
                    ;46873          ;1-------------------------------;
                    ;46874          M[TEMP7]_MB.AND.ZLIT8[OFF],     ; ISOLATE 2ND BYTE FOR TESTING OSR.
U 170D, 0186,7D92,0037,FC47,0170,E  ;46875          VA_VA+4                         ;
                    ;46876
                    ;46877 =0       ;0-------------------------------;
                    ;46878          WRITE M[TEMP1],SIZE[LONG],      ; WRITE PATTERN FOR XB<63:32> TO CACHE.
U 170E, 0080,1592,4020,05D8,057B,A  ;46879          PUSH,NEXT/MV.FILL.XB            ; LET XB PRE-FETCH COMPLETELY.
                    ;46880
                    ;46881          ;1-------------------------------;
                    ;46882          VA_XB.XOR.R[TEMP0] PC_PC+4,     ; CHECK XB<31:0>
U 170F, 0081,7017,4A20,24A7,0171,0  ;46883          WX.EQ.0?                        ;
                    ;46884
                    ;46885 =0       ;0-------------------------------;
U 1710, 0980,0C37,0037,8C87,003E,F  ;46886          PC_ZLIT0[OF1],NEXT/MV.ERROR     ; ERROR IN XB<31:0>
                    ;46887
                    ;46888          ;1-------------------------------;
                    ;46889          VA_XB.XOR.R[TEMP1] PC_PC+4,     ; CHECK XB<63:32>
U 1711, 0C81,7017,4A20,64A7,0171,2  ;46890          WX.EQ.0?                        ;
                    ;46891
                    ;46892 =0       ;0-------------------------------;
U 1712, 0980,0C37,0037,9487,003E,F  ;46893          PC_ZLIT0[OF2],NEXT/MV.ERROR     ;ERROR IN XB<63:32>
```

```
                                        ;46894          ;1-------------------------------;
U 1713, 0580,0C37,0030,0487,017B,7      ;46895          PC_ZLIT0[0]                       ; TO WAIT FOR XB TO FINISH PRE-FETCHING
                                        ;46896
                                        ;46897          ;-------------------------------; NEXT ADDRESS MUST BE IN 0-1K
U 17B7, 0480,0036,4170,0047,0018,F      ;46898          IRD1TEST                          ; LOAD IR AND OSR
                                        ;46899
                                        ;46900 .REGION/IRD1.R1L,IRD1.R1H
                                        ;46901 =1111
                                        ;46902          ;-------------------------------;
U 018F, 0C80,0036,4030,0567,017B,8      ;46903          MDR_ZEXT(IR)                      ;
                                        ;46904
                                        ;46905 .REGION/MV.R1L,MV.R1H/MV.R2L,MV.R2H/MV.R3L,MV.R3H
                                        ;46906          ;-------------------------------;
                                        ;46907          VA_M[MDR]-R[TEMP0],               ; CHECK IR AGAINST TEST PATTERN<7:0>
U 17B8, 0881,2000,0B40,04A7,0954,6 374* ;46908          SIGND CMP?,SIZE[BYTE]             ;
                                        ;46909
                                        ;46910 =10      ;10-----------------------------; ERROR IN IR
U 1546, 0580,0C37,0037,A487,003E,F      ;46911          PC_ZLIT0[0F4],NEXT/MV.ERROR       ;
                                        ;46912
                                        ;46913          ;11-----------------------------;
                                        ;46914          MDR_ZEXT(OSR),                    ;
U 1547, 0C84,02B7,0001,C5E7,017B,9      ;46915          R[TEMP7]_RB.RR.8                  ; ROTATE TEST PATTERN<15:8> TO <7:0>
                                        ;46916
                                        ;46917          ;-------------------------------;
                                        ;46918          VA_M[MDR].XOR.R[TEMP7],           ; CHECK OSR AGAINST TEST PATTERN<15:8>
U 17B9, 0081,2003,4A31,C4A7,0171,4      ;46919          WX.EQ.0?                          ;
                                        ;46920
                                        ;46921 =0       ;0-----------------------------; ERROR IN OSR
U 1714, 0580,0C37,0037,BC87,003E,F      ;46922          PC_ZLIT0[0F7],NEXT/MV.ERROR       ;
                                        ;46923
                                        ;46924          ;1-------------------------------;
                                        ;46925          PC_ZLIT0[0],                      ; ZERO FOR MV.XB.PC.TEST
U 1715, 0980,0C37,00B0,0487,0000,2      ;46926          RETURN [+2]                       ; TEST OK
```

```
                                ;46927 .TOC "  MICRO VERIFY                    :       MV.FILL.XB"
                                ;46928
                                ;46929 ;*************************************************************************
                                ;46930 ;
                                ;46931 ;       Input           VA              Must be between 3 and 20
                                ;46932 ;
                                ;46933 ;       Output          VA              0
                                ;46934 ;                       PC              0
                                ;46935 ;                       XB              Cache from VA to VA+7
                                ;46936 ;
                                ;46937 ;       This subroutine is used to point the PC to zero, and wait for the
                                ;46938 ;       XB to finish pre-fetching, this is done to make sure all 64 bits of
                                ;46939 ;       the XB are tested and not just the first 32 bits.
                                ;46940 ;*************************************************************************
                                ;46941
                                ;46942 MV.FILL.XB:
                                ;46943          ;-------------------------------;
U 17BA, 0C80,0036,4030,04E7,017B,8      ;46944          MDR_0                           ; STALL FOR CACHE WRITE TO FINISH.
                                ;46945
                                ;46946          ;-------------------------------;
U 17BB, 0D80,0C37,0030,0487,0171,6      ;46947          PC_ZLIT0[0]                     ; POINT PC TO TEST PATTERN TO FILL XB
                                ;46948 =0
                                ;46949 MV.XB.WAIT:
                                ;46950          ;0-------------------------------;
                                ;46951          VA_M[VA]-ZLIT0[1],              ; WAIT FOR XB TO FINISH PRE-FETCHING
U 1716, 0981,BC10,0A30,0CA7,0971,6 342* ;46952          WX.EQ.0?,NEXT/MV.XB.WAIT        ;
                                ;46953
                                ;46954          ;1-------------------------------;
U 1717, 0880,0036,4080,0047,0000,1      ;46955          RETURN [+1]                     ;
```

```
                                ;46956 .TOC "  MICRO VERIFY                        : XB, PC_PC+ISIZE TEST"
                                ;46957
                                ;46958 ;*****************************************************************************
                                ;46959 ;
                                ;46960 ;       Resources       LONLIT          Used to generate test patterns
                                ;46961 ;                       TEMP0           Test pattern for 1st 32 bits of istream
                                ;46962 ;                       TEMP1           Test pattern for 2nd 32 bits of istream
                                ;46963 ;                       VA              Used to load test patterns into cache.
                                ;46964 ;
                                ;46965 ;       Output          PC              Error code if error exit.
                                ;46966 ;                       VA              Result of bad compare if error exit.
                                ;46967 ;
                                ;46968 ;       Subroutines     MV.CHECK.XB
                                ;46969 ;
                                ;46970 ;       This section tests the auto incrementing of the PC when fetching istream
                                ;46971 ;       data.  A byte, a word, a longword, and then another byte are fetched
                                ;46972 ;       from the XB and checked, letting PC be incremented automatically.
                                ;46973 ;*****************************************************************************
                                ;46974
                                ;46975 MV.XB.PC.TEST:
                                ;46976          ;-------------------------------;
U 17BC, 0380,02AD,5667,8047,017B,D ;46977       LONLIT_[0AA55330F]              ; LOAD TEST PATTERN
                                ;46978
                                ;46979          ;-------------------------------;
U 17BD, 0486,05BE,403D,4047,017B,E ;46980       M[TEMP0]_R[LONLIT]              ; LOAD 32 BITS OF TES PATTERN
                                ;46981
                                ;46982          ;-------------------------------;
U 17BE, 0286,1E7F,C03D,4047,0152,8 ;46983       M[TEMP1]_R[LONLIT].NOT          ; LOAD SECOND 32 BITS OF TEST PATTERN
                                ;46984
                                ;46985 =0*      ;0*-----------------------------; LOAD PATTERN INTO XB.  THE CHECK HERE
U 1528, 0C80,0036,4030,0047,0570,C ;46986       PUSH,NEXT/MV.CHECK.XB           ; IS A SIDE EFFECT OF LOADING.
                                ;46987
                                ;46988          ;1*-----------------------------;
                                ;46989          VA_ZEXT(XB)-R[TEMP0] PC_PC+1,   ; CHECK SOURCING OF 1 BYTE
U 152A, 0081,7014,0B40,24A7,0971,8 385* ;46990  SIGND CMP?                       ;
                                ;46991
                                ;46992 =0       ;0-------------------------------;
U 1718, 0980,0C37,0038,8C87,003E,F ;46993       PC_ZLIT0[111],NEXT/MV.ERROR     ; ERROR SOURCING FIRST BYTE
                                ;46994
                                ;46995          ;1-------------------------------;
U 1719, 0C84,03B7,0000,0047,017B,F ;46996       R[TEMP0]_M[TEMP0].RR.8          ; SHIFT PATTERN FOR CHECKING.
                                ;46997
                                ;46998          ;-------------------------------;
                                ;46999          VA_ZEXT(XB)-R[TEMP0] PC_PC+2,   ; CHECK INCREMENT OF PC BY 1 AND
U 17BF, 0481,7014,0B50,24A7,0958,6 385* ;47000  SIGND CMP?                       ; SOURCING OF A WORD.
                                ;47001
                                ;47002 =10      ;10------------------------------; ERROR SOURCING 2 BYTES OR
U 1586, 0980,0C37,0038,9487,003E,F ;47003       PC_ZLIT0[110],NEXT/MV.ERROR     ; ERROR INCREMENTING PC BY 1.
                                ;47004
                                ;47005          ;11------------------------------;
U 1587, 0F80,02A9,987A,A847,017C,0 ;47006       LONLIT_[0AACCF0AA]              ; LOAD CHECK PATTERN
```

```
                                        ;47007          ;--------------------------------;
                                        ;47008          VA_XB.XOR.R[LONLIT] PC_PC+4,     ; CHECK INCREMENT OF PC BY 2 AND
U 1700, 0081,7017,4A2D,64A7,0171,A      ;47009          WX.EQ.0?                         ; SOURCING OF AN UNALIGNED LONGWORD.
                                        ;47010
                                        ;47011 =0       ;0------------------------------; ERROR SOURCING AN UNALIGNED LONGWORD
) 171A, 0580,0C37,0038,A487,003E,F      ;47012          PC_ZLITO[114],NEXT/MV.ERROR      ; OR ERROR INCREMENTING PC BY 2.
                                        ;47013
                                        ;47014          ;1------------------------------;
U 171B, 0081,729F,4000,24A7,017C,1      ;47015          VA_ZEXT(XB).XOR.(R[TEMP0].RR.8) PC_PC+1 ; CHECK INCREMENT OF PC BY 4
                                        ;47016
                                        ;47017          ;--------------------------------;
                                        ;47018          VA_ZEXT(M[VA]),SIZE[BYTE],       ; CHECK RESULT OF PREVIOUS COMPARE
U 17C1, 0881,B016,4A0D,84A7,0171,C      ;47019          WX.EQ.0?                         ;
                                        ;47020
                                        ;47021 =0       ;0------------------------------;
) 171C, 0580,0C37,0038,BC87,003E,F      ;47022          PC_ZLITO[117],NEXT/MV.ERROR      ; ERROR INCREMENTING PC BY 4
```

```
                        ;47023 .TOC "  MICRO VERIFY                    : DSIZE TEST"
                        ;47024
                        ;47025 ;*****************************************************************************
                        ;47026 ;
                        ;47027 ;       Input           VA              Must be zero.
                        ;47028 ;
                        ;47029 ;       Resources       RNUM            Used to address MTEMPs.
                        ;47030 ;                       DREG            Used to load RNUM when loading MTEMPs.
                        ;47031 ;                       MTEMP0,1,2,4,8  Used to check loading of RNUM.
                        ;47032 ;                       TEMP3           Used to hold instruction dependent size.
                        ;47033 ;
                        ;47034 ;       Output          PC              Error code if error exit.
                        ;47035 ;                       VA              Result of bad compare if error exit.
                        ;47036 ;
                        ;47037 ;       This section uses the fact that the DSIZE ROM for the HALT instruction
                        ;47038 ;       is filled with BYTE, WORD, LONG, QUAD, WORD, LONG, to test that the
                        ;47039 ;       ROM is properly accessed and that the DSIZE latch is working.
                        ;47040 ;       It also tests that RNUM gets loaded with the register number portion
                        ;47041 ;       of the operand specifier when OSR is loaded.  This is all done by
                        ;47042 ;       loading IR with an IRD1TEST and then loading OSR and RNUM using
                        ;47043 ;       BUT/LOD.INC.BRA which will grab a byte at a time from the istream and
                        ;47044 ;       load the DSIZE latch with the next size out of the DSIZE ROM.
                        ;47045 ;*****************************************************************************
                        ;47046
                        ;47047 MV.DSIZE.TEST:
                        ;47048          ;1-------------------------------;
U 171D, 0B80,046C,F57F,F847,0171,E      ;47049          LONLIT_[72615000]               ; LOAD TEST PATTERN
                        ;47050
                        ;47051 =0       ;0-------------------------------;
                        ;47052          WRITE R[LONLIT],SIZE[LONG],     ; WRITE HALT AND FIRST 3 OPERAND SPECS.
U 171E, 0880,05BE,402D,45D8,057D,D      ;47053          PUSH,NEXT/MV.STALL.FOR.CACHE    ;
                        ;47054
                        ;47055          ;1-------------------------------;
U 171F, 0780,07FA,FB3B,DC47,017C,2      ;47056          LONLIT_[00A09884],VA_VA+4       ; LOAD REMAINING TEST PATTERN
                        ;47057
                        ;47058          ;-------------------------------;
U 1702, 0080,05BE,402D,45D8,017C,3      ;47059          WRITE R[LONLIT],SIZE[LONG]      ; WRITE REMAINING 3 OPERAND SPECIFIERS
                        ;47060
                        ;47061          ;-------------------------------;
U 1703, 0D81,DC37,2030,44E7,017C,4      ;47062          D_RNUM_ZLIT0[8],MDR_0           ;
                        ;47063
                        ;47064 MV.DSIZE.SETUP:
                        ;47065          ;-------------------------------; (PC WILL GET 0 WHEN DONE)
U 1704, 0887,05B2,4A30,0487,0972,0 322* ;47066          PC_M[TEMP.R]_D,WX.EQ.0?         ; LOAD MTEMPS FOR CHECKING RNUM
                        ;47067
                        ;47068 =0       ;0-------------------------------;
                        ;47069          D_RNUM_D-ZLIT0[1],              ; DECREMENT RNUM AND LOOP
U 1720, 0181,DC30,2030,0847,017C,4      ;47070          NEXT/MV.DSIZE.SETUP             ;
                        ;47071
                        ;47072          ;1-------------------------------; NEXT ADDRESS MUST BE IN 0-1K
U 1721, 0C80,0036,4170,0047,0019,F      ;47073          IRD1TEST                        ; LOAD IR AND OSR
                        ;47074
                        ;47075 .REGION/IRD1.R1L,IRD1.R1H
                        ;47076 =1111    ;1111----------------------------;
U 019F, 0480,0036,4030,0047,0173,2      ;47077          NEXT/MV.DSIZE.FIRST             ; LOAD IR WITH 0, OSR WITH 50.
```

```
                                ;47078 .REGION/MV.R1L,MV.R1H/MV.R2L,MV.R2H/MV.R3L,MV.R3H
                                ;47079 =1111
                                ;47080 MV.LOAD.DSIZE:
                                ;47081          ;1111-----------------------------;
                                ;47082          M[TEMP3]_CONX.SIZ,SIZE[IDEP],    ;
U 16EF, 0C86,3737,00B0,0047,0000,1  ;47083      RETURN [+1]                      ; GET INST. DEPENDENT SIZE FROM ROM.
                                ;47084 =0
                                ;47085 MV.DSIZE.FIRST:
                                ;47086          ;0-----------------------------; IRD COUNT IS 0
U 1722, 0C80,0036,4030,0047,056E,F  ;47087      PUSH,NEXT/MV.LOAD.DSIZE          ; LOAD DSIZE ROM (BYTE) INTO TEMP3.
                                ;47088
                                ;47089          ;1-----------------------------;
                                ;47090          VA_M[TEMP3].XOR.ZLIT0[1],        ; CHECK IF DSIZE IS BYTE
U 1723, 0D80,3C13,4A30,0CA7,0172,4  ;47091      WX.EQ.0?                         ;
                                ;47092
                                ;47093 =0       ;0-----------------------------;
U 1724, 0980,0C37,0039,0C87,003E,F  ;47094      PC_ZLIT0[121],NEXT/MV.ERROR      ; ERROR READING DSIZE ROM.
                                ;47095
                                ;47096          ;-----------------------------;
                                ;47097          VA_M[TEMP.R].XOR.ZLIT0[0],       ; CHECK THAT RNUM IS 0.
U 1725, 0D81,0C13,4A30,04A7,0166,4  ;47098      WX.EQ.0?                         ;
                                ;47099
                                ;47100 =00      ;00----------------------------;
U 1664, 0980,0C37,0039,1487,003E,F  ;47101      PC_ZLIT0[122],NEXT/MV.ERROR      ; ERROR LOADING RNUM.
                                ;47102
                                ;47103          ;01----------------------------; IRD COUNT IS 1
                                ;47104          LOD INC BRA?,                    ; LOAD OSR WITH 61, REG DEFERRED R1
U 1665, 0C80,0036,41B0,0047,056E,F  ;47105      PUSH,NEXT/MV.LOAD.DSIZE          ; AND LOAD DSIZE ROM (WORD) INTO TEMP3.
                                ;47106
                                ;47107          ;10----------------------------;
                                ;47108          VA_M[TEMP3].XOR.ZLIT0[2],        ; CHECK IF DSIZE IS WORD
U 1666, 0580,3C13,4A30,14A7,0172,6  ;47109      WX.EQ.0?                         ;
                                ;47110 =
                                ;47111 =0       ;-----------------------------;
U 1726, 0580,0C37,0039,2487,003E,F  ;47112      PC_ZLIT0[124],NEXT/MV.ERROR      ; ERROR READING DSIZE ROM.
```

```
                                ;47113 .TOC "  MICRO VERIFY                    : PARITY CHECKERS TEST"
                                ;47114
                                ;47115 ;*****************************************************************************
                                ;47116 ;
                                ;47117 ;       Resources       FPDOFFSET       Used to get control back from mach check
                                ;47118 ;                       VA              Used for addressing cache and TB.
                                ;47119 ;                       TEMP1           Temp for memory status/control regs.
                                ;47120 ;
                                ;47121 ;       Output          PC              Error code if error exit.
                                ;47122 ;                       VA              Result of bad compare if error exit.
                                ;47123 ;
                                ;47124 ;       This section uses special micro code functions to write bad parity in
                                ;47125 ;       the Cache and in both sides of the Translation BUffer. The location with
                                ;47126 ;       the bad parity is then referenced and a micro trap should occur.
                                ;47127 ;       FPDOFFSET is setup so the machine check micro code will return to this
                                ;47128 ;       test and then the memory/status bits are checked to verify that the
                                ;47129 ;       error was properly detected. Also a special control store location
                                ;47130 ;       with bad parity is referenced which should get micro trapped with
                                ;47131 ;       the test again regaining control because of FPDOFFSET.
                                ;47132 ;*****************************************************************************
                                ;47133 MV.PARITY.TEST:
                                ;47134         ;1-----------------------------; **** MAY NOT BE NEEDED ??????
                                ;47135         M[FPDOFFSET]_ZLIT28[18],        ; SETUP TO RETURN FROM MACHINE CHECK.
U 1727, 0106,CC77,0030,C047,017C,5 ;47136      CLEAR FLAGO                     ; CLEAR FLAGO FOR CACHE PARITY.
                                ;47137
                                ;47138         ;-----------------------------; **** MAY NOT BE NEEDED ??????
U 17C5, 0D80,0C37,0030,04A7,017C,6 ;47139      VA_ZLIT0[0]                     ; BE SURE VA HAS A GOOD ADDRESS
```

```
                                ;47140 .TOC "  MICRO VERIFY                    :       CACHE PARITY TEST"
                                ;47141
                                ;47142 MV.CACHE.PARITY:
                                ;47143          ;-------------------------------;
U 17C6, 0CF8,0036,4030,05D8,017C,7 ;47144       WRITE,FORCE CACHE PARITY        ;
                                ;47145
                                ;47146          ;-------------------------------; HOLD TO ALLOW CACHE WRITE TO COMPLETE
U 17C7, 04F8,0036,4030,04E7,017C,8 ;47147       MDR_0,FORCE CACHE PARITY        ;
                                ;47148
                                ;47149          ;-------------------------------;
U 17C8, 0880,0036,4020,0050,0167,4 ;47150       READ,SIZE[LONG]                 ;
                                ;47151
                                ;47152 =00      ;00-----------------------------;
                                ;47153          PC_ZLIT0[181],MB_M[MDR],        ; SOURCE MDR TO FORCE MACHINE CHECK.
U 1674, 0981,2C37,003C,0C87,003E,F ;47154       NEXT/MV.ERROR                   ; THIS MICRO WORD SHOULD GET TRAPPED.
                                ;47155
                                ;47156          ;01-----------------------------; LOAD ADDRESS OF MACHINE CHECK
                                ;47157          MEMSCAR_ZLIT24[8],              ; ERROR SUMMARY REGISTER.
U 1675, 0580,0CB7,0030,4687,057D,4 ;47158       PUSH,NEXT/MV.GET.CLEAR.MEMSCR   ; READ IT INTO TEMP1 AND CLEAR IT.
                                ;47159
                                ;47160          ;10-----------------------------; MACHINE CHECK ERROR SUMMARY REGISTER
                                ;47161          VA_M[TEMP1].XOR.ZLIT24[08],     ; SHOULD SAY BUS ERROR.
U 1676, 0580,1C93,4A30,44A7,0167,8 ;47162       WX.EQ.0?                        ;
                                ;47163 =
                                ;47164 =00      ;00-----------------------------;
U 1678, 0980,0C37,003C,1487,003E,F ;47165       PC_ZLIT0[182],NEXT/MV.ERROR     ; BAD MACHINE CHECK ERROR SUMMARY REG.
                                ;47166
                                ;47167          ;01-----------------------------;
                                ;47168          MEMSCAR_ZLIT24[4],              ; LOAD ADDRESS OF CACHE ERROR REG.
U 1679, 0980,0CB7,0030,2687,057D,4 ;47169       PUSH,NEXT/MV.GET.CLEAR.MEMSCR   ; READ IT INTO TEMP1 AND CLEAR IT.
                                ;47170
                                ;47171          ;10-----------------------------; CACHE ERROR REGISTER
                                ;47172          VA_M[TEMP1].XOR.ZLIT24[0D],     ; SHOULD SAY TAG AND DATA ERROR AND HIT.
U 167A, 0980,1C93,4A30,6CA7,0172,C ;47173       WX.EQ.0?                        ;
                                ;47174 =
                                ;47175 =0       ;0------------------------------;
U 172C, 0580,0C37,003C,2487,003E,F ;47176       PC_ZLIT0[184],NEXT/MV.ERROR     ; BAD CACHE ERROR REGISTER.
```

```
                                ;47177 .TOC "  MICRO VERIFY                    :       TB GROUP 0 PARITY TEST"
                                ;47178
                                ;47179 MV.TB.GRP0.PARITY:
                                ;47180          ;1-------------------------------;
U 172D, 0980,0CB7,0030,0687,017C,9  ;47181      MEMSCAR_ZLIT24[0]               ; LOAD ADDRESS OF MME BIT.
                                ;47182
                                ;47183          ;-------------------------------;
U 17C9, 0C80,0E77,0030,0607,017C,A  ;47184      MEMSCR_-1                       ; SET MME.
                                ;47185
                                ;47186          ;-------------------------------;
U 17CA, 0180,0CB7,0030,1E87,017C,B  ;47187      MEMSCAR_ZLIT24[3]               ; LOAD ADDRESS OF TB DISABLE REG.
                                ;47188
                                ;47189          ;-------------------------------;
U 17CB, 0180,0CB7,0030,5607,017C,C  ;47190      MEMSCR_ZLIT24[0A]               ; FORCE REPLACEMENT OF GROUP 0.
                                ;47191
                                ;47192          ;-------------------------------;
U 17CC, 04F1,BE77,0030,0507,0168,0  ;47193      TB_BAD PARITY                   ;
                                ;47194
                                ;47195 =00      ;00-------------------------------;
                                ;47196          READ,SIZE[LONG],                ; CAUSE TB PARITY ERROR.
U 1680, 0580,0C37,002D,8C90,017D,2  ;47197      PC_ZLIT0[1B1],NEXT/MV.CLEAR.MME ; THIS MICRO WORD SHOULD GET TRAPPED.
                                ;47198
                                ;47199          ;01-------------------------------;
                                ;47200          MEMSCAR_ZLIT24[0D],             ; LOAD ADDRESS OF TB GROUP PARITY REG.
U 1681, 0980,0CB7,0030,6E87,057D,4  ;47201      PUSH,NEXT/MV.GET.CLEAR.MEMSCR   ; READ IT INTO TEMP1 AND CLEAR IT.
                                ;47202
                                ;47203          ;10-------------------------------;
U 1682, 0481,B5BE,403D,8507,017C,D  ;47204      TB_R[ZERO]                      ; CLEAR THE ERROR IN THE TB.
                                ;47205 =
                                ;47206          ;-------------------------------; TB GROUP ERROR REGISTER
                                ;47207          VA_M[TEMP1].XOR.ZLIT24[05],     ; SHOULD SAY TAG AND DATA ERROR GROUP 0.
U 17CD, 0D80,1C93,4A30,2CA7,0168,8  ;47208      WX.EQ.0?                        ;
                                ;47209
                                ;47210 =00      ;00-------------------------------;
U 1688, 0180,0C37,003D,9487,017D,2  ;47211      PC_ZLIT0[1B2],NEXT/MV.CLEAR.MME ; BAD TB GROUP PARITY ERROR REGISTER.
                                ;47212
                                ;47213          ;01-------------------------------; LOAD ADDRESS OF MACHINE CHECK
                                ;47214          MEMSCAR_ZLIT24[8],              ; ERROR SUMMARY REGISTER.
U 1689, 0580,0CB7,0030,4687,057D,4  ;47215      PUSH,NEXT/MV.GET.CLEAR.MEMSCR   ; READ IT INTO TEMP1 AND CLEAR IT.
                                ;47216
                                ;47217          ;10-------------------------------; MACHINE CHECK ERROR SUMMARY REGISTER
                                ;47218          VA_M[TEMP1].XOR.ZLIT24[04],     ; SHOULD SAY TB ERROR.
U 168A, 0180,1C93,4A30,24A7,0172,E  ;47219      WX.EQ.0?                        ;
                                ;47220 =
                                ;47221 =0       ;0--------------------------------;
U 172E, 0D80,0C37,003D,A487,017D,2  ;47222      PC_ZLIT0[1B4],NEXT/MV.CLEAR.MME ; BAD MACHINE CHECK ERROR SUMMARY REG.
```

```
                                ;47223 .TOC "  MICRO VERIFY                    :       TB GROUP 1 PARITY TEST"
                                ;47224
                                ;47225 MV.TB.GRP1.PARITY:
                                ;47226          ;1------------------------------;
U 172F, 0180,0CB7,0030,1E87,017C,E ;47227       MEMSCAR_ZLIT24[3]               ; LOAD ADDRESS OF TB DISABLE REG.
                                ;47228
                                ;47229          ;------------------------------;
U 17CE, 0980,0CB7,0030,6E07,017C,F ;47230       MEMSCR_ZLIT24[0D]               ; FORCE REPLACEMENT OF GROUP 1.
                                ;47231
                                ;47232          ;------------------------------;
U 17CF, 0CF1,BE77,0030,0507,0169,0 ;47233       TB_BAD PARITY                   ;
                                ;47234
                                ;47235 =00      ;00------------------------------;
                                ;47236          READ,SIZE[LONG],                ; CAUSE TB PARITY ERROR.
U 1690, 0980,0C37,002D,BC90,017D,2 ;47237       PC_ZLIT0[1B7],NEXT/MV.CLEAR.MME ; THIS MICRO WORD SHOULD GET TRAPPED.
                                ;47238
                                ;47239          ;01------------------------------;
                                ;47240          MEMSCAR_ZLIT24[0D],             ; LOAD ADDRESS OF TB GROUP PARITY REG.
U 1691, 0980,0CB7,0030,6E87,057D,4 ;47241       PUSH,NEXT/MV.GET.CLEAR.MEMSCR   ; READ IT INTO TEMP1 AND CLEAR IT.
                                ;47242
                                ;47243          ;10------------------------------;
U 1692, 0481,B5BE,403D,8507,017D,0 ;47244       TB_R[ZERO]                      ; CLEAR ERROR IN TB.
                                ;47245 =
                                ;47246          ;------------------------------; TB GROUP ERROR REGISTER
                                ;47247          VA_M[TEMP1].XOR.ZLIT24[0A],     ; SHOULD SAY TAG AND DATA ERROR GROUP 1.
U 17D0, 0980,1C93,4A30,54A7,0169,4 ;47248       WX.EQ.0?                        :
                                ;47249
                                ;47250 =00      ;00------------------------------;
U 1694, 0180,0C37,003D,C487,017D,2 ;47251       PC_ZLIT0[1B8],NEXT/MV.CLEAR.MME ; BAD TB GROUP PARITY REGISTER.
                                ;47252
                                ;47253          ;01------------------------------; LOAD ADDRESS OF MACHINE CHECK
                                ;47254          MEMSCAR_ZLIT24[8],              ; ERROR SUMMARY REGISTER.
U 1695, 0580,0CB7,0030,4687,057D,4 ;47255       PUSH,NEXT/MV.GET.CLEAR.MEMSCR   ; READ IT INTO TEMP1 AND CLEAR IT.
                                ;47256
                                ;47257          ;10------------------------------; MACHINE CHECK ERROR SUMMARY REGISTER
                                ;47258          VA_M[TEMP1].XOR.ZLIT24[04],     ; SHOULD SAY TB ERROR.
U 1696, 0180,1C93,4A30,24A7,0173,0 ;47259       WX.EQ.0?                        :
                                ;47260 =
                                ;47261 =0       ;0-------------------------------;
U 1730, 0180,0C37,003D,DC87,017D,2 ;47262       PC_ZLIT0[1BB],NEXT/MV.CLEAR.MME ; BAD MACHINE CHECK ERROR SUMMARY REG.
                                ;47263
                                ;47264          ;------------------------------;
U 1731, 0980,0CB7,0030,0687,017D,1 ;47265       MEMSCAR_ZLIT24[0]               ; LOAD ADDRESS OF MME
                                ;47266
                                ;47267          ;------------------------------;
U 17D1, 0080,05B7,0030,0607,017F,D ;47268       MEMSCR_0,NEXT/MV.CS.PARITY      ; CLEAR MME, END OF TB TEST.
```

```
                                ;47269 MV.CLEAR.MME:
                                ;47270          ;--------------------------------;
U 17D2, 0180,0CB7,0030,0687,017D,3 ;47271       MEMSCAR_ZLIT24[0]               ; LOAD ADDRESS OF MME
                                ;47272
                                ;47273          ;--------------------------------;
U 17D3, 0080,05B7,0030,0607,003E,F ;47274       MEMSCR_0,NEXT/MV.ERROR          ;
                                ;47275
                                ;47276 MV.GET.CLEAR.MEMSCR:
                                ;47277          ;--------------------------------;
U 17D4, 0886,1036,4030,0647,017D,5 ;47278       M[TEMP1]_MEMSCR                 ; SAVE MEMORY STATUS REGISTER.
                                ;47279
                                ;47280          ;--------------------------------;
U 17D5, 0D86,1C92,0030,7847,017D,6 ;47281       M[TEMP1]_MB.AND.ZLIT24[0F]      ; CLEAR UNDEFINED BITS.
                                ;47282
                                ;47283          ;--------------------------------;
U 17D6, 0480,05B7,00B0,0607,0000,1 ;47284       MEMSCR_0,RETURN [+1]            ; CLEAR MEMORY STATUS REGISTER.
                                ;47285
                                ;47286
                                ;47287
                                ;47288 .TOC "  MICRO VERIFY                     :       CONTROL STORE PARITY TEST"
                                ;47289
                                ;47290 17FD:    ;FORCE FOR DIAGNOSTICS.....;
                                ;47291 MV.CS.PARITY:
                                ;47292          ;0--------------------------------; *******BAD PARITY*************
                                ;47293          PC_ZLIT0[1D1],                  ;
                                ;47294          PAR1/1,PAR2/0,                  ; FORCE PARITY
U 17FD, 0580,0C37,003E,8C87,003E,F ;47295       NEXT/MV.ERROR                   ; THIS WORD SHOULD GET MICRO TRAPPED.
                                ;47296
                                ;47297 17FE:    ;1--------------------------------;
                                ;47298          VA_M[ERRCOD].XOR.ZLIT0[1],      ; CHECK FOR CONTROL STORE PARITY ERROR
U 17FE, 0180,BC13,4A30,0CA7,0173,2 ;47299       WX.EQ.0?                        ;
                                ;47300
                                ;47301 =0       ;0--------------------------------;
U 1732, 0980,0C37,003E,9487,003E,F ;47302       PC_ZLIT0[1D2],NEXT/MV.ERROR     ; NOT CONTROL STORE PARITY ERROR.
```

```
                                        ;47303 .TOC "  MICRO VERIFY                    : CACHE TEST"
                                        ;47304
                                        ;47305 ;*****************************************************************************
                                        ;47306 ;
                                        ;47307 ;       Resources       LONLIT          Constant 7FC (highest cache address).
                                        ;47308 ;                       TEMP1           Temp for generating cache addresses.
                                        ;47309 ;                       Q               Temp to generate test patterns.
                                        ;47310 ;                       MDR             Contents of tested cache location.
                                        ;47311 ;                       STEPC           Used to loop setting 1 bit at a time.
                                        ;47312 ;
                                        ;47313 ;       Output          PC              Error code if error exit.
                                        ;47314 ;
                                        ;47315 ;       This section does a "stuck at" and address test of the cache by
                                        ;47316 ;       clearing the entire cache, then filling the entire cache 1 bit
                                        ;47317 ;       at a time, checking each word for zero before writing it and checking
                                        ;47318 ;       after setting each bit.
                                        ;47319 ;*****************************************************************************
                                        ;47320
                                        ;47321 MV.CACHE.TEST:
                                        ;47322          ;1-----------------------------;
U 1733, 0380,07FF,FF80,1847,017D,7      ;47323          LONLIT_[0FFC]                   ; LOAD ADDRESS OF HIGHEST CACHE LOCATION
                                        ;47324
                                        ;47325          ;-----------------------------;
U 17D7, 0886,15BE,403D,44A7,0169,9      ;47326          VA_M[TEMP1]_R[LONLIT]           ; LOAD VA FOR CLEARING CACHE.
                                        ;47327 =00
                                        ;47328 =01
                                        ;47329 MV.CLEAR.CACHE:
                                        ;47330          ;01----------------------------;
                                        ;47331          WRITE R[ZERO],SIZE[LONG],       ; CLEAR CACHE.
U 1699, 0880,05BE,402D,85D8,057D,D      ;47332          PUSH,NEXT/MV.STALL.FOR.CACHE    ;
                                        ;47333
                                        ;47334 MV.CLEAR.CACHE.LOOP:
                                        ;47335          ;10----------------------------;
                                        ;47336          VA_M[TEMP1]_MB-ZLIT0[4],        ; DECREMENT VA TO
                                        ;47337          SIGND CMP?,SIZE[LONG],          ;
U 169A, 0586,1C10,0B60,24A7,0969,9 413* ;47338          NEXT/MV.CLEAR.CACHE             ; CLEAR THE CACHE IN REVERSE ORDER.
                                        ;47339
                                        ;47340          ;11----------------------------;
U 169B, 0086,15BE,403D,44A7,016A,1      ;47341          VA_M[TEMP1]_R[LONLIT]           ; LOAD ADDRESS OF HIGHEST CACHE LOCATION
                                        ;47342 =01
                                        ;47343 MV.FILL.CACHE:
                                        ;47344          ;01----------------------------;
                                        ;47345          READ,SIZE[LONG],Q_0,            ; READ TEST LOCATION, CLEAR Q FOR
U 16A1, 0080,05B7,1020,0050,017D,8      ;47346          NEXT/MV.CACHE.CHECK.ZERO        ; GENERATING FILL BITS.
                                        ;47347
                                        ;47348          ;11----------------------------;
U 16A3, 0CA0,0036,4030,0047,0173,6      ;47349          STEPC_2,NEXT/MV.END             ; LOAD STEPC FOR CALL TO CN.TYPE.CHAR
```

```
                                    ;47350 MV.CACHE.CHECK.ZERO:
                                    ;47351          ;-------------------------------;
                                    ;47352          WB_M[MDR],                      ; LOCATION SHOULD BE ZERO.
U 17D8, 0881,2592,4A30,0047,096A,8 322* ;47353      WX.EQ.0?                        ;
                                    ;47354
                                    ;47355 =00      ;00-----------------------------;
U 16A8, 0980,0C37,003F,0C87,003E,F  ;47356          PC_ZLIT0[1E1],NEXT/MV.ERROR     ;
                                    ;47357
                                    ;47358          ;01-----------------------------;
                                    ;47359          WRITE Q (Q.SL.1).OR.1,          ; FILL LOCATION 1 BIT AT A TIME.
                                    ;47360          SIZE[LONG],                     ;
U 16A9, 0C80,0139,D02D,85D8,057D,D  ;47361          PUSH,NEXT/MV.STALL.FOR.CACHE    ;
                                    ;47362
                                    ;47363 MV.READ.CACHE:
                                    ;47364          ;10-----------------------------;
                                    ;47365          READ,SIZE[LONG],                ;
U 16AA, 0880,0036,4020,0050,017D,A  ;47366          NEXT/MV.CACHE.CHECK.FILL        ; READ THE TEST LOCATION.
                                    ;47367
                                    ;47368          ;11-----------------------------;
                                    ;47369          VA_M[TEMP1]_MB-ZLIT0[4],        ; DECREMENT VA TO FILL AND TEST
                                    ;47370          SIGND CMP?,SIZE[LONG],          ;
U 16AB, 0D86,1C10,0B60,24A7,096A,1 413* ;47371      NEXT/MV.FILL.CACHE              ; THE CACHE IN REVERSE ORDER.
                                    ;47372
                                    ;47373 MV.CACHE.CHECK.FILL:
                                    ;47374          ;-------------------------------;
U 17DA, 0881,200B,4A30,0047,0173,4  ;47375          WB_M[MDR].XOR.Q,WX.EQ.0?        ; CHECK EACH TIME A BIT IS WRITTEN.
                                    ;47376
                                    ;47377 =0       ;0-----------------------------;
U 1734, 0980,0C37,003F,1487,003E,F  ;47378          PC_ZLIT0[1E2],NEXT/MV.ERROR     ;
                                    ;47379
                                    ;47380          ;1-----------------------------;
                                    ;47381          WRITE Q (Q.SL.1).OR.1,          ; FILL TEST LOCATION 1 BIT AT A TIME.
U 1735, 0C80,0139,D02D,85D8,017D,C  ;47382          SIZE[LONG]                      ;
                                    ;47383
                                    ;47384          ;-------------------------------;
                                    ;47385          MDR_0,                          ; ALLOW CACHE WRITE TO FINISH.
U 17DC, 0480,0036,4330,04E7,016A,A  ;47386          DBZ~STEPC?,NEXT/MV.READ.CACHE   ; LOOP UNTIL ALL 32 BITS ARE SET.
```

```
                                ;47387 MV.STALL.FOR.CACHE:
                                ;47388          ;-------------------------------;
U 17DD, 0080,0036,40B0,04E7,0000,1  ;47389          MDR_0,RETURN [+1]               ;
                                ;47390 3EF:
                                ;47391 MV.ERROR:
                                ;47392          ;*********FORCE ADDRESS*********; GENERATE AN OFFSET TO ADD TO THE ERROR
U 03EF, 01A6,5C37,0034,0047,016B,0  ;47393          M[TEMP5]_ZLIT0[80],STEPC_2      ; CODE TO MAKE A PRINTABLE CHARACTER.
                                ;47394
                                ;47395 =00      ;00-------------------------------;
                                ;47396          R[TEMP5].SIZ_(RB+M[PC].ASR.3).SR.1, ;
                                ;47397          DEC STEPC,                      ; STEPC MUST BE 1.
U 16B0, 009B,A77D,8001,4047,049F,4  ;47398          PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;47399
                                ;47400          ;01-------------------------------;
                                ;47401          MEMSCAR_ZLIT24[0E],             ; LOAD ADDRESS OF CMI DISABLE REGISTER.
U 16B1, 0980,0CB7,0030,7687,04BF,E  ;47402          PUSH,NEXT/MV.CLEANUP            ;
                                ;47403
                                ;47404          ;10-------------------------------;
                                ;47405          M[FPDOFFSET]_ZLIT0[OFF],        ; SET ERROR CODE FOR CONSOLE HALT.
U 16B2, 0986,CC37,0037,F847,0000,1  ;47406          NEXT/MS.HALT.MICRO              ;
                                ;47407 =
                                ;47408 =0
                                ;47409 MV.END:
                                ;47410          ;0-------------------------------;
                                ;47411          M[TEMP5]_ZLIT0[25],DEC STEPC,   ; TYPE % TO INDICATE SUCCESS.
U 1736, 059E,5C37,0031,2847,049F,4  ;47412          PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;47413
                                ;47414          ;1-------------------------------;
U 1737, 0180,0CB7,0030,7687,00BF,E  ;47415          MEMSCAR_ZLIT24[0E]              ; LOAD ADDRESS OF CMI DISABLE REGISTER.
                                ;47416 0BFE:
                                ;47417 MV.CLEANUP:
                                ;47418          ;*********FORCE ADDRESS*********;
U 0BFE, 0080,05B7,0030,0607,016B,4  ;47419          MEMSCR_0                        ; ENABLE THE CMI.
                                ;47420
                                ;47421 =00      ;00-------------------------------;
                                ;47422          M[TEMP5]_ZLIT0[0D],             ; TYPE <CR><LF>.
U 16B4, 0586,5C37,0030,6847,049F,4  ;47423          PUSH,NEXT/CN.TYPE.CHAR          ;
                                ;47424
                                ;47425          ;01-------------------------------;
U 16B5, 0480,0036,4030,0047,0484,D  ;47426          PUSH,NEXT/IN.CLR.CACHE.ROUT     ; CLEAR THE CACHE.
                                ;47427
                                ;47428 MV.SET.CRAR:
                                ;47429          ;10-------------------------------;
                                ;47430          CRAR_ZLIT16[0C0],               ; LOAD ADDRESS FOR SETTING HALT.
U 16B6, 0518,0D37,00B6,0247,0000,1  ;47431          CLEAR FLAG3,RETURN [+1]         ; END OF TEST, RETURN TO CALLER.
                                ;47432 =
```

```
                                        ;47433 .TOC             "GHROM.MIC"
                                        ;47434 .TOC             "REVISION 00.05"
                                        ;47435 ;       Audrey R.Reith
                                        ;47436
                                        ;47437
;47438  .NOBIN
;47439  .TOC    "       Revision History"
;47440
;47441  ;REV    EXPLANATION
;47442
;47443  ;       ADDED COMMENTS TO LISTING
;47444  ; 00    RELEASE OF THE SUBROUTINES FOR MAIN CONTROL ROM
;47445  ;       FOR GH FLOAT INSTRUCTIONS IN WCS
;47446
                                        ;47447 .BIN
```

```
                        ;47448 .TOC     "       GH FLOAT Subroutines in ROM     : Generalized routines FX.CCS"
                        ;47449 ;*****************************************************************************
                        ;47450 ;
                        ;47451 ;       FX.CCS
                        ;47452 ;               This subroutine set the floating point ccs for most
                        ;47453 ;       of the GFLOAT instructions (CVTS and Arithmetics)
                        ;47454 ;       Bits 0-31 of the destination have already been written.
                        ;47455 ;       Exit is to FX.WRITE.SEC to increment
                        ;47456 ;       VA and to write bits 32-63. Note FLAG1 must be clear if
                        ;47457 ;       bits 32-63 are in Temp3 or set if in MDR.
                        ;47458 ;
                        ;47459 ;
                        ;47460 ;*****************************************************************************
                        ;47461 .REGION/GHFLT.R1L,GHFLT.R1H/GHFLT.R2L,GHFLT.R2H/GHFLT.R3L,GHFLT.R3H
                        ;47462
                        ;47463 =00
                        ;47464 FX.CCS:
                        ;47465          ;00-----------------------------; RESERVED OPERAND
U 0788, 0C80,0036,4030,0047,00FF,8      ;47466          NEXT/IE.OPER.FAULT              ;
                        ;47467
                        ;47468          ;01-----------------------------; VALUE IS NEGATIVE
                        ;47469          CC_ZLITO[8],                    ;
U 0789, 0108,0C37,0030,40A7,0056,F      ;47470          CLEAR FLAG1,NEXT/FX.WRITE.SEC   ; DEFINED IN FLOAT.MIC
                        ;47471
                        ;47472          ;10-----------------------------; VALUE IS 0
                        ;47473          CC_ZLITO[4],                    ;
U 078A, 0D08,0C37,0030,20A7,0056,F      ;47474          CLEAR FLAG1,NEXT/FX.WRITE.SEC   ; DEFINED IN FLOAT.MIC
                        ;47475
                        ;47476          ;11-----------------------------; VALUE IS POSITIVE
                        ;47477          CC_ZLITO[0],                    ;
U 078B, 0508,0C37,0030,00A7,0056,F      ;47478          CLEAR FLAG1,NEXT/FX.WRITE.SEC   ; DEFINED IN FLOAT.MIC
```

```
                        ;47479 .TOC     ""      GH FLOAT Subroutines in ROM     : Generalized routines FX.MULSIGN''
                        ;47480 ;*****************************************************************************
                        ;47481 ;
                        ;47482 ;       FX.MULSIGN
                        ;47483 ;               This subroutine sets D and MTemp8 to 0/8000 and FLAG0 =0/1
                        ;47484 ;       for positive/negative sign of the result for DIVG,MULG,EMODG
                        ;47485 ;
                        ;47486 ;       Input:
                        ;47487 ;               FLAG0 and FLAG1 = sign of operands 1 and 2
                        ;47488 ;
                        ;47489 ;*****************************************************************************
                        ;47490 =00
                        ;47491 FX.MULSIGN:
                        ;47492         ;00-----------------------------; ANSWER IS POSITIVE
                        ;47493         M[TEMP8] D_ZLIT0[0],            ; SIGN BIT FOR ANSWER IN BIT 15
                        ;47494         CLEAR ADD1?(FLAG0),             ; ALSO USE FLAG 0 FOR SIGN OF ANSWER
J 078C, 0906,8C37,20B0,0047,0000,2      ;47495         RETURN [+2]                     ; RETS +2  !!!
                        ;47496
                        ;47497         ;01-----------------------------; ANSWER IS NEGATIVE
                        ;47498         M[TEMP8] D_ZLIT12[8],           ; SIGN BIT FOR ANSWER IN BIT 15
                        ;47499         SET ADD1?F[AG0),                ; ALSO USE FLAG 0 FOR SIGN OF ANSWER
J 078D, 0546,8D77,20B0,4047,0000,2      ;47500         RETURN [+2]                     ; RETS +2  !!!
                        ;47501
                        ;47502         ;10-----------------------------; ANSWER IS NEGATIVE
                        ;47503         M[TEMP8] D_ZLIT12[8],           ; SIGN BIT FOR ANSWER IN BIT 15
                        ;47504         SET ADD1?F[AG0),                ; ALSO USE FLAG 0 FOR SIGN OF ANSWER
J 078E, 0546,8D77,20B0,4047,0000,2      ;47505         RETURN [+2]                     ; RETS +2  !!!
                        ;47506
                        ;47507         ;11-----------------------------; ANSWER IS POSITIVE
                        ;47508         M[TEMP8] D_ZLIT0[0],            ; SIGN BIT FOR ANSWER IN BIT 15
                        ;47509         CLEAR ADD1?(FLAG0),             ; ALSO USE FLAG 0 FOR SIGN OF ANSWER
J 078F, 0906,8C37,20B0,0047,0000,2      ;47510         RETURN [+2]                     ; RETS +2  !!!
                        ;47511
                        ;47512
                        ;47513 FX.SL11P4:
                        ;47514         ;-------------------------------;
J 0?77, 0D80,0F?6,4030,5847,0077,9      ;47515         SL_[11.]                        ; FOR SHARING WITH DIFFERENT RTNS
                        ;47516
                        ;47517 FX.PL4:
                        ;47518         ;-------------------------------;
                        ;47519         PL_[4],                         ;
J 0?79, 0D80,0EF6,40B0,2047,0000,1      ;47520         RETURN [+1]                     ;
```

```
                        ;47521 .TOC     ""      GH FLOAT Subroutines in ROM     : Generalized routines FX.CVTS.CC""
                        ;47522 ;*****************************************************************************
                        ;47523 ;
                        ;47524 ;       FX.CVTS.CC
                        ;47525 ;               This subroutine sets the cc for integer results in CVTG/H
                        ;47526 ;       to integer types and it tests for integer overflow.
                        ;47527 ;       Input:
                        ;47528 ;               FLAG2 = if set is integer overflow
                        ;47529 ;
                        ;47530 ;       FX.INT.OVR.TST
                        ;47531 ;               This entry is also used by EMODG and EMODH after all
                        ;47532 ;       results have been written to determine if integer overflow
                        ;47533 ;       has occurred and if so whether or not to honor the trap depending
                        ;47534 ;       on the IV bit in the PSL. This is integer overflow -a TRAP condition
                        ;47535 ;*****************************************************************************
                        ;47536 =00
                        ;47537 FX.CVTS.CC:
                        ;47538         ;00-------------------------------; >0 VALUE POSITIVE
                        ;47539         CC_ZLITO[0],                    ;
                        ;47540         OVER(FLAG2)?,                   ; TEST INTEGER OVERFLOW
U 0790, 0580,0C37,04B0,00A7,0078,0      ;47541         NEXT/FX.INT.OVR.TST             ;
                        ;47542
                        ;47543         ;01-------------------------------; VALUE = 0
                        ;47544         CC_ZLITO[4],                    ;
                        ;47545         OVER(FLAG2)?,                   ; TEST INTEGER OVERFLOW
U 0791, 0D80,0C37,04B0,20A7,0078,0      ;47546         NEXT/FX.INT.OVR.TST             ;
                        ;47547
                        ;47548         ;10-------------------------------; VALUE IS NEGATIVE
                        ;47549         CC_ZLITO[8],                    ;
                        ;47550         OVER(FLAG2)?,                   ; TEST INTEGER OVERFLOW
U 0792, 0180,0C37,04B0,40A7,0078,0      ;47551         NEXT/FX.INT.OVR.TST             ;
                        ;47552 =
                        ;47553 =0*0
                        ;47554 FX.INT.OVR.TST:
                        ;47555         ;0*0-------------------------------; NOT INTEGER OVERFLOW
U 0780, 0080,0036,4130,0047,003F,9      ;47556         IRD1                            ; FINISHED AN INSTRUCTION
                        ;47557
                        ;47558         ;0*1-------------------------------; USED FOR AFTER TEST V
                        ;47559         M[ERRCOD]_ZLITO[1],             ;
                        ;47560         SET FLAG3,                      ;
U 0781, 0D5E,BC37,0030,0847,00F8,4      ;47561         NEXT/IE.ARITH.TRP               ; OFF THE MAIN TRAP CODE
                        ;47562
                        ;47563 =1*0
                        ;47564 FX.INT.OVR:
                        ;47565         ;1*0-------------------------------; USED WHEN TESTING FLAG2
U 0784, 0880,0036,4030,08E7,0077,D      ;47566         SET V                           ; MUST TEST PSL BIT FOR TRAP
                        ;47567 =
                        ;47568 FX.TSTV.PSL:
                        ;47569         ;-------------------------------;
U 077D, 0486,0036,4030,0087,0078,5      ;47570         M[TEMP0]_PSL                    ; GET PSL
                        ;47571
                        ;47572         ;-------------------------------;
                        ;47573         WB_M[TEMP0].AND.ZLITO[20],      ; IS INTEGER OVERFLOW ENABLED
                        ;47574         WX.NE.0?,                       ;
U 0785, 0980,0C12,0A71,0047,0078,0      ;47575         NEXT/FX.INT.OVR.TST             ;
```

```
                        ;47576 .TOC     ''      GH FLOAT Subroutines in ROM     : Parse routines FX.GOPER1.10,FX.GOP
                        ;47577 ;*****************************************************************************
                        ;47578 ;
                        ;47579 ;       These routines used to parse GFLOAT operands into exp and fraction
                        ;47580 ;       and to test for reserved operands.
                        ;47581 ;       Input: for both FX.GOPER1.10 and FX.GOPER2.10
                        ;47582 ;              POP1C(FLAG4) - determines which routine =0/1 respectively
                        ;47583 ;              PLATCH = 4,SLATCH = 11
                        ;47584 ;       For FX.GOPER1.10
                        ;47585 ;         Input
                        ;47586 ;              Temp1,Temp3 = packed operand
                        ;47587 ;         Output:
                        ;47588 ;              Temp5 = exponent biased
                        ;47589 ;              Temp1,Temp3 = fraction with overflow and hidden bit
                        ;47590 ;              PLATCH changed to 22
                        ;47591 ;              FLAG0 = sign of operand
                        ;47592 ;              FLAG2 = operand = 0 condition
                        ;47593 ;       For FX.GOPER2.10
                        ;47594 ;         Input
                        ;47595 ;              Temp1,Temp4 = packed operand
                        ;47596 ;         Output:
                        ;47597 ;              Temp7 = exponent biased
                        ;47598 ;              Temp1,Temp4 = fraction with overflow and hidden bit
                        ;47599 ;              PLATCH changed to 22
                        ;47600 ;              FLAG1 = sign of operand
                        ;47601 ;              FLAG3 = operand = 0 condition
                        ;47602 ;       For FX.GOPER.POLY.ARG
                        ;47603 ;         Input and Output same as above except
                        ;47604 ;              RTemp8 gets the exponent
                        ;47605 ;
                        ;47606 ;       FX.TYPE.HID
                        ;47607 ;              used by FX.HOP1 and FX.HOP2 for deciding if overflow
                        ;47608 ;       position to be present in fraction
                        ;47609 ;*****************************************************************************
                        ;47610 =0
                        ;47611 FX.GOPER1.10:
                        ;47612          ;0-------------------------------; 1ST OPERAND TYPE
                        ;47613          R[TEMP5]_M[TEMP1].XZ,           ; EXP TO T5 PL/SL=4/11.
                        ;47614          FRO.FLTZ?,                      ;
U 077E, 0C84,1077,0AB1,4047,0879,4 335* ;47615          NEXT/FX.GOPER1.20               ; VALIDATE AND CHECK SIGN
                        ;47616
                        ;47617 FX.GOPER2.10:
                        ;47618          ;1-------------------------------; 2ND OPERAND TYPE
                        ;47619          R[TEMP7]_M[TEMP1].XZ,           ; EXP TO T7 PL/SL=4/11.
                        ;47620          FRO.FLTZ?,                      ;
U 077F, 0884,1077,0AB1,C047,0879,8 335* ;47621          NEXT/FX.GOPER2.20               ; VALIDATE AND CHECK SIGN
```

```
                                ;47622 .TOC     ''      GH FLOAT Subroutines in ROM     : Parse routines FX.GOPER1.20''
                                ;47623 =00
                                ;47624 FX.GOPER1.20:
                                ;47625          ;00-------------------------------; RESERVED OPERAND
U 0794, 0C80,0036,4030,0047,00FF,8  ;47626          NEXT/IE.OPER.FAULT              ;
                                ;47627
                                ;47628          ;01-------------------------------; VALUE < 0
                                ;47629          M[TEMP1]_MB.RR.16,              ; ROT TO GET FRACTION EXTRACTED
                                ;47630          SET FLAG0,                      ; SET SIGN FLAG
U 0795, 0846,13B7,0010,0047,0079,F  ;47631          NEXT/FX.GOPER12                 ; CONTINUE ON GETTING FRACTION
                                ;47632
                                ;47633          ;10-------------------------------; VALUE = 0
                                ;47634          M[TEMP1]_0,                     ; ZERO OUT WORKING TEMPS
                                ;47635          SET FLAG2,                      ; FLAG =0
U 0796, 0C56,15B7,0030,0047,0054,2  ;47636          NEXT/FX.ZEROTEMP3               ; ZERO SECOND HALF AND RETURN
                                ;47637
                                ;47638          ;11-------------------------------; VALUE > 0
                                ;47639          M[TEMP1]_MB.RR.16,              ; ROT TO GET FRACTION EXTRACTED
                                ;47640          CLEAR FLAG0,                    ; CLEAR SIGN FLAG
U 0797, 0006,13B7,0010,0047,0079,F  ;47641          NEXT/FX.GOPER12                 ; CONTINUE ON GETTING FRACTION
                                ;47642
                                ;47643 =0
                                ;47644 FX.TYPE.HID:
                                ;47645          ;0-------------------------------; FOR H FLOAT
                                ;47646          M[TEMP1]_MB.OR.ZLIT24[80],      ; MERGE ONLY
U 07B4, 0586,1C92,40B4,0047,0000,1  ;47647          RETURN [+1]                     ;
                                ;47648
                                ;47649 FX.HIDBIT:
                                ;47650          ;1-------------------------------; FOR G FLOAT
U 07B5, 0D86,1E92,0039,F847,0079,3  ;47651          M[TEMP1]_MB.AND.OLIT24[13F]     ; EXTRACT AND MERGE
                                ;47652
                                ;47653          ;-------------------------------;
                                ;47654          M[TEMP1]_MB.OR.ZLIT28[4],       ;
U 0793, 0986,1C52,40B0,2047,0000,1  ;47655          RETURN [+1]                     ;
                                ;47656
                                ;47657 FX.GOPER12:
                                ;47658          ;-------------------------------;
                                ;47659          PL_[22.],                       ; SET UP FOR FRACTION EXTRACT
                                ;47660          POP1C(FLAG4)?,                  ; WHICH ONE
U 079F, 0180,0EF6,4470,B047,007B,6  ;47661          NEXT/FX.GOPER1.30               ;
                                ;47662
                                ;47663 =0
                                ;47664 FX.GOPER1.30:
                                ;47665          ;0-------------------------------
                                ;47666          M[TEMP1]_(MB R[TEMP3]).RR.P,    ; GET BITS 0-31
U 07B6, 0486,1137,0030,C047,007B,5  ;47667          NEXT/FX.HIDBIT                  ; ADD HIDDEN AND OVERFLOW BITS
                                ;47668
                                ;47669          ;1-------------------------------; SECCND OPERAND TYPE
                                ;47670          M[TEMP1]_(MB R[TEMP4]).RR.P,    ; GET BITS 0-31
U 07B7, 0086,1137,0031,0047,007B,5  ;47671          NEXT/FX.HIDBIT                  ; ADD HIDDEN AND OVERFLOW BITS
```

```
                                ;47672 .TOC   ''      GH FLOAT Subroutines in ROM     : Parse routines FX.GOPER2.20''
                                ;47673 =00
                                ;47674 FX.GOPER2.20:
                                ;47675          ;00-----------------------------; RESERVED OPERAND
U 0798, 0C80,0036,4030,0047,00FF,8  ;47676          NEXT/IE.OPER.FAULT              ;
                                ;47677
                                ;47678          ;01-----------------------------; VALUE NEGATIVE
                                ;47679          M[TEMP1]_MB.RR.16,              ; ROT TO GET FRACTION EXTRACTED
                                ;47680          SET FLAG1,                      ; SET SIGN FLAG
U 0799, 004E,13B7,0010,0047,0079,F  ;47681          NEXT/FX.GOPER12                 ; CONTINUE ON GETTING FRACTION
                                ;47682
                                ;47683          ;10-----------------------------; VALUE IS ZERO
                                ;47684          M[TEMP4]_0,                     ; ZERO OUT WORKING TEMP
                                ;47685          SET FLAG3,                      ; SET OPERAND IS 0 FLAG
U 079A, 045E,45B7,0030,0047,006F,0  ;47686          NEXT/FX.ZEROTEMP1               ; ZERO OUT BITS 0-31 RETURN
                                ;47687
                                ;47688          ;11-----------------------------; VALUE POSITIVE
                                ;47689          M[TEMP1]_MB.RR.16,              ; ROT TO GET FRACTION EXTRACTED
                                ;47690          CLEAR FLAG1,                    ;
U 079B, 080E,13B7,0010,0047,0079,F  ;47691          NEXT/FX.GOPER12                 ; CONTINUE ON GETTING FRACTION
                                ;47692
                                ;47693 FX.GOPER.POLY.ARG:
                                ;47694          ;-----------------------------;
                                ;47695          R[TEMP8]_M[TEMP1].XZ,           ; ARG ENTRY
                                ;47696          FRO.FLTZ?,                      ; EXP IN RT8, PL/SL=4/11.
U 07A3, 0884,1077,0AB2,0047,0879,8 335*  ;47697          NEXT/FX.GOPER2.20               ; VALIDATE AND CHECK SIGN
```

```
                                ;47698 .TOC     "       GH FLOAT Subroutines in ROM     : Normalization  FX.NORM.GFL "
                                ;47699 ;*****************************************************************************
                                ;47700 ;
                                ;47701 ;       Normalization and packing routines for GFLOAT instructions
                                ;47702 ;       If enter at FX.NORM.GFL
                                ;47703 ;         Input:
                                ;47704 ;               Temp2,Temp3 = unnormalized fraction
                                ;47705 ;               Temp5 = biased exponent with fudge factor for PLATCH
                                ;47706 ;               D = 0/8000 depending on sign of result
                                ;47707 ;       The fraction is normalized and the hidden bit removed. The value
                                ;47708 ;       is then rounded (800 position in Temp3) and the carry if
                                ;47709 ;       present propagated through the most significant bits and into
                                ;47710 ;       the exponent.  The value of PLATCH is added to the exponent
                                ;47711 ;       and the result is tested for under or overflow .  If not
                                ;47712 ;       detected the packing operation is undertaken with SLATCH set to 12
                                ;47713 ;       and the value of D merged into the result.
                                ;47714 ;
                                ;47715 ;*****************************************************************************
                                ;47716 =0
                                ;47717 FX.NORM.GFL:
                                ;47718         ;0-------------------------------; ENTRY TO NORMALIZATION ROUTINE
                                ;47719         PL_MSS M[TEMP2],                 ; ANY BIT SET IN BITS 0-31
                                ;47720         CLEAR FLAG3,                     ; BE SURE FLAG CLEAR
                                ;47721         WX.NE.0?,                        ;
U 07B8, 0018,29C2,4A7D,8047,0079,C ;47722         NEXT/FX.NORM.GFL.10              ; CONTINUE ON
                                ;47723
                                ;47724         ;1-------------------------------; INT OR TIMER TEST ENTRY
                                ;47725         PUSH,INTPEND OR TIMER?,          ; RETS -1 IF TIMER ONLY
U 07B9, 0480,0036,4AF0,0047,04F7,0 ;47726         NEXT/IE.SERV.IP.TS2              ;
                                ;47727
                                ;47728 =00
                                ;47729 FX.NORM.GFL.10:
                                ;47730         ;00------------------------------; NO 1 BIT SET IN 0-31
                                ;47731         PL_MSS M[TEMP3],                 ; CHECK BITS 32-63
                                ;47732         WX.NE.0?,                        ;
U 079C, 0480,39C2,4A7D,8047,007A,4 ;47733         NEXT/FX.NORM.GFL.50              ; CONTINUE LOOKING
                                ;47734
                                ;47735         ;01------------------------------; EXTRACT FRACTION FROM BITS 0-31
                                ;47736         M[TEMP2]_(MB R[TEMP3]).RR.P,     ; GETS RID OF HIDDEN BIT TOO
U 079D, 0C86,2137,0030,C047,046E,E ;47737         PUSH,NEXT/FX.T3ZERO              ; GET BITS 32-63
                                ;47738
                                ;47739         ;10------------------------------;
U 079E, 0D86,3D91,0030,4047,007B,A ;47740         M[TEMP3]_MB+ZLIT8[8]             ; ROUND FRACTION (800)
                                ;47741 =
                                ;47742 =0
                                ;47743         ;0-------------------------------;
                                ;47744         M[TEMP2]_MB+R[ZERO]+ALKC,        ; PROPAGATE CARRY INTO BITS 0-31
U 07BA, 0486,2041,003D,8047,045C,D ;47745         PUSH,NEXT/FX.NORM.ADD.ALKC       ; ADD TO EXPONENT
                                ;47746
                                ;47747 FX.NORM.GFL.20:
                                ;47748         ;1-------------------------------;
                                ;47749         M[TEMP5]_MB+PL,                  ; ADD FACTOR TO EXPONENT
                                ;47750         SIZE[WORD],                      ;
                                ;47751         SIGND CMP?,                      ; LOOK FOR UNDER OR OVERFLOW
U 07BB, 0086,5B11,0B50,0047,087A,0 410* ;47752         NEXT/FX.NORM.GFL.30              ;
```

```
                                        ;47753 .TOC   "      GH FLOAT Subroutines in ROM     : Normalization  FX.NORM.GFL "
                                        ;47754 =00
                                        ;47755 FX.NORM.GFL.30:
                                        ;47756          ;00-----------------------------; > 0 MUST CHECK FOR OVERFLOW
                                        ;47757          WB_M[TEMP5].AND.ZLIT8[0F8],     ;
                                        ;47758          WX.NE.0?,                       ;
U 07A0, 0180,5D92,0A77,C047,007B,E      ;47759          NEXT/FX.PACK.GFL.15             ;
                                        ;47760
                                        ;47761          ;01-----------------------------; =0 UNDERFLOW
                                        ;47762          R[TEMP2]_0,                     ; CLEAR OUT BITS 0-31
U 07A1, 0884,05B7,0030,8047,005C,E      ;47763          NEXT/FX.TESTPSL                 ; SEE IF FAULT ON UNDERFLOW
                                        ;47764
                                        ;47765          ;10-----------------------------; <0 UNDERFLOW
                                        ;47766          R[TEMP2]_0,                     ; IF UNDERFLOW FAULT ENABLE NO RETURN
U 07A2, 0884,05B7,0030,8047,005C,E      ;47767          NEXT/FX.TESTPSL                 ; OTHERWISE RET TO CALLING RTN
                                        ;47768 =
                                        ;47769 =00
                                        ;47770 FX.NORM.GFL.50:
                                        ;47771          ;00-----------------------------; NO BITS SET ANS IS 0
                                        ;47772          R[TEMP2]_0,                     ; BOTH WORDS ZERO
U 07A4, 0C84,05B7,00B0,8047,0000,1      ;47773          RETURN [+1]                     ;
                                        ;47774
                                        ;47775          ;01-----------------------------; BIT FOUND IN BITS 32-63
                                        ;47776          M[TEMP3]_(MB R[ZERO]).RR.P,     ; EXTRACT FROM BITS 32-63
U 07A5, 0886,3137,003D,8047,046B,D      ;47777          PUSH,NEXT/FX.T3TOT2             ; EXTRACT T3 INTO T2 AND T3 =0
                                        ;47778
                                        ;47779          ;10-----------------------------; ADJUST EXP
                                        ;47780          M[TEMP5]_MB-ZLIT0[20],          ; FOR THE EXTRA 32 BITS
                                        ;47781          SIZE[WORD],                     ;
U 07A6, 0586,5C10,0011,0047,007B,B      ;47782          NEXT/FX.NORM.GFL.20             ; NO NEED TO ROUND
                                        ;47783 =
```

```
                                ;47784 .TOC     ""      GH FLOAT Subroutines in ROM     : Packing FX.PACK.GFL""
                                ;47785 =0
                                ;47786 FX.PACK.GFL:
                                ;47787          ;0------------------------------;
                                ;47788          R[TEMP2] (M[TEMP5] RB).RR.S,    ; EXTRACT BITS 0-31
U 07BC, 0084,51F7,0030,8047,007A,8      ;47789          NEXT/FX.PACK.GFL.30             ; CONTINUE
                                ;47790
                                ;47791 FX.PACK.GFL.10:
                                ;47792          ;1------------------------------;
U 07BD, 0084,21F7,0030,C047,007A,7      ;47793          R[TEMP3] (M[TEMP2] RB).RR.S     ; POSITON BITS 32-63
                                ;47794
                                ;47795 FX.PACK.GFL.12:
                                ;47796          ;------------------------------;
                                ;47797          M[TEMP3] MB.RR.16,              ; ROT BITS 32-63
U 07A7, 0486,33B7,0010,0047,007B,C      ;47798          NEXT/FX.PACK.GFL                ; CONTINUE WITH BITS 0-31
                                ;47799
                                ;47800
                                ;47801 =0
                                ;47802 FX.PACK.GFL.15:
                                ;47803          ;0------------------------------; NO OVERFLOW
                                ;47804          SL [12.],                       ; SET UP FOR PACKING
U 07BE, 0D80,0F76,4030,6047,007B,D      ;47805          NEXT/FX.PACK.GFL.10             ;
                                ;47806
                                ;47807          ;1------------------------------; OVERFLOW
                                ;47808          VA M[VA]-ZLIT0[4],              ; DO THIS FOR POLYG
                                ;47809          SET FLAG1,                      ;
U 07BF, 0549,BC10,0030,24A7,00FF,B      ;47810          NEXT/IE.FLOV.FAULT              ; OFF TO FAULT HANDLER
                                ;47811
                                ;47812 FX.PACK.GFL.30:
                                ;47813          ;------------------------------;
                                ;47814          M[TEMP2] D.OR.(MB.RR.16),       ; MERGE BITS 0-31 WITH SIGN IN D
U 07A8, 0886,23B2,4090,0047,0000,1      ;47815          RETURN [+1]                     ;
```

```
                                ;47816 .TOC    ''       GH FLOAT Subroutines in ROM     : Generalized routines ''
                                ;47817 ;*****************************************************************************
                                ;47818 ;
                                ;47819 ;
                                ;47820 ;       SOME OTHER SHORT SUBROUTINES
                                ;47821 ;
                                ;47822 ;
                                ;47823 ;*****************************************************************************
                                ;47824 .REGION/GHFLT.R1L,GHFLT.R1H/GHFLT.R2L,GHFLT.R2H/GHFLT.R3L,GHFLT.R3H
                                ;47825
                                ;47826 FX.ADJTMPS.LFT2:
                                ;47827          ;-------------------------------;
U 07B0, 0884,3592,4030,8047,007C,6  ;47828      R[TEMP2]_M[TEMP3]               ; MOVE DATA DOWN
                                ;47829
                                ;47830 FX.ADJTMPS.LFT:
                                ;47831          ;-------------------------------;
U 07C6, 0084,4592,4030,C047,007C,7  ;47832      R[TEMP3]_M[TEMP4]               ; MOVE DATA DOWN
                                ;47833
                                ;47834          ;-------------------------------;
U 07C7, 0084,5592,4031,0047,007C,8  ;47835      R[TEMP4]_M[TEMP5]               ;
                                ;47836
                                ;47837 FX.T5ZERO:
                                ;47838          ;-------------------------------;
                                ;47839          R[TEMP5]_0,                     ;
U 07C8, 0084,05B7,00B1,4047,03FF,F  ;47840      RETURN [-1]                     ; RETS -1 !!!!!!
                                ;47841
                                ;47842 FX.LFTSHFT3:
                                ;47843          ;-------------------------------;
U 07C9, 0C86,3137,0031,0047,007C,A  ;47844      M[TEMP3]_(MB R[TEMP4]).RR.P     ; LEFT SHIFT THE DATA
                                ;47845
                                ;47846 FX.LFTSHFT4:
                                ;47847          ;-------------------------------;
U 07CA, 0486,4137,0031,4047,007C,B  ;47848      M[TEMP4]_(MB R[TEMP5]).RR.P     ;
                                ;47849
                                ;47850          ;-------------------------------;
                                ;47851          M[TEMP5]_(MB R[ZERO]).RR.P,     ;
U 07CB, 0C86,5137,00BD,8047,0000,1  ;47852      RETURN [+1]                     ;
                                ;47853
                                ;47854 FX.PROP.CRY:
                                ;47855          ;-------------------------------;
U 07CC, 0486,4041,003D,8047,007C,D  ;47856      M[TEMP4]_MB+R[ZERO]+ALKC        ; CARRY FROM ROUND POSITION
                                ;47857
                                ;47858          ;-------------------------------;
U 07CD, 0086,3041,003D,8047,007C,E  ;47859      M[TEMP3]_MB+R[ZERO]+ALKC        ;
                                ;47860
                                ;47861          ;-------------------------------;
U 07CE, 0C86,2041,003D,8047,007C,F  ;47862      M[TEMP2]_MB+R[ZERO]+ALKC        ;
                                ;47863
                                ;47864          ;-------------------------------;
                                ;47865          M[TEMP8]_MB+R[ZERO]+ALKC,       ; INTO EXP
                                ;47866          SIZE[LONG],                     ;
                                ;47867          ALUS_SIGND,                     ; SAVE SIGN VALUE
U 07CF, 0486,8041,00AD,98C7,0000,1  ;47868      RETURN [+1]                     ;
```

```
                                ;47869 .TOC    ''       GH FLOAT Subroutines in ROM     : Generalized routines ''
                                ;47870
                                ;47871 FX.ADJTMPS.RT5:
                                ;47872          ;--------------------------------;
U 07D0, 0884,4137,0031,4047,007D,1  ;47873          R[TEMP5]_(M[TEMP4] RB).RR.P     ; EXTRACT THE RIGHT PARTS
                                ;47874
                                ;47875 FX.ADJTMPS.RT4:
                                ;47876          ;--------------------------------;
U 07D1, 0884,3137,0031,0047,007D,2  ;47877          R[TEMP4]_(M[TEMP3] RB).RR.P     ; SHIFT VALUE TO THE RIGHT
                                ;47878
                                ;47879 FX.ADJTMPS.RT3:
                                ;47880          ;--------------------------------;
                                ;47881          R[TEMP3]_(M[TEMP2] RB).RR.P,    ;
U 07D2, 0C84,2137,00B0,C047,0000,1  ;47882          RETURN [+1]                     ;
                                ;47883
                                ;47884 FX.ROTTMP.T5T3:
                                ;47885          ;--------------------------------;
U 07D3, 0C86,53B7,0010,0047,007D,4  ;47886          M[TEMP5]_MB.RR.16               ; ROT THE TEMP
                                ;47887
                                ;47888          ;--------------------------------;
U 07D4, 0086,43B7,0010,0047,007D,5  ;47889          M[TEMP4]_MB.RR.16               ;
                                ;47890
                                ;47891          ;--------------------------------;
                                ;47892          M[TEMP3]_MB.RR.16,              ;
U 07D5, 0886,33B7,0090,0047,0000,1  ;47893          RETURN [+1]                     ;
                                ;47894
                                ;47895
                                ;47896
                                ;47897 FX.PLOSL15:
                                ;47898          ;--------------------------------;
U 07D6, 0180,0EF6,4030,0047,007D,7  ;47899          PL_[0]                          ;
                                ;47900
                                ;47901 FX.SL15:
                                ;47902          ;--------------------------------;
                                ;47903          SL_[15.],                       ;
U 07D7, 0980,0F76,40B0,7847,0000,1  ;47904          RETURN [+1]                     ;
                                ;47905
                                ;47906 FX.ZERORT12:
                                ;47907          ;--------------------------------;
                                ;47908          R[TEMP12]_0,                    ;
U 07D8, 0884,05B7,00B3,0047,0000,1  ;47909          RETURN [+1]                     ;
```

```
                                ;47910 .TOC     "       GH FLOAT Subroutines in ROM     : HFLOAT to Integer FX.CVTFI.HSUB.20
                                ;47911 ;*****************************************************************************
                                ;47912 ;
                                ;47913 ;       This subroutine converts HFLOAT type operands to integer(CVTH to x
                                ;47914 ;       and EMODH). The initial test for Exp>=0 is performed in GHWCS.MIC
                                ;47915 ;       Only if the exp is representable in 7 bits (less than 128) is
                                ;47916 ;       this routine entered to find the integer bits of significance
                                ;47917 ;       and to window the fraction correctly all other cases have 0s for
                                ;47918 ;       integer and fraction parts.
                                ;47919 ;       Input:
                                ;47920 ;               MTemp8 - exponent unbiased
                                ;47921 ;               Temp0  - constant of 32  incremented by 32 for each iteration
                                ;47922 ;               STEPC  - 6  for # tries for positioning low order bits
                                ;47923 ;               Temp2-5 - fraction with bit 31 of T2 most significant bit
                                ;47924 ;               MDR - 0
                                ;47925 ;               FLAG2 - 0
                                ;47926 ;               FLAG4 - to determine if round or truncate (1/0 respectively)
                                ;47927 ;       Output:
                                ;47928 ;               MDR = integer
                                ;47929 ;               T2-T5 = fraction positioned with most significant  bit 31 of T2
                                ;47930 ;               FLAG2 = used to signify integer overflow
                                ;47931 ;*****************************************************************************
                                ;47932 07C0:
                                ;47933 FREE.07C0:
                                ;47934         ;0------------------------------; EXP <0 NO INTEGER
                                ;47935         M[TEMP8]_MB-ZLIT8[40],          ;
                                ;47936         CLEAR FLAG2,                    ;
U 07C0, 0D16,8D90,0472,0047,007C,2 ;47937      FLAG4?,NEXT/FX.CVTFI.HSUB.110   ; IS ROUND BIT CLEARED OR NOT
                                ;47938
                                ;47939 07C1:
                                ;47940 FREE.07C1:
                                ;47941         ;1------------------------------; EXP >=0
                                ;47942         M[TEMP8]_MB-ZLIT8[40],          ; REMOVE BIAS
U 07C1, 0D86,8D90,0032,0047,007D,9 ;47943      NEXT/FX.CVTFI.HSUB.10           ;
                                ;47944
                                ;47945 FX.CVTFI.HSUB.10:
                                ;47946         ;------------------------------;
U 07D9, 0116,0C37,0031,0047,007A,9 ;47947      M[TEMP0]_ZLIT0[20],CLEAR FLAG2  ; CONSTANT IN TEMP (32.)
                                ;47948 =00
                                ;47949 =01
                                ;47950 FX.CVTFI.HSUB.20:
                                ;47951         ;01------------------------------;
                                ;47952         PL_R[TEMP0]-M[TEMP8],           ; RANGE OF EXP
                                ;47953         EXPONENT RANGE?,                ;
U 07A9, 0480,8BC3,0DF0,0047,087A,C 377* ;47954 NEXT/FX.CVTFI.HSUB.50           ;
                                ;47955
                                ;47956 FX.CVTFI.HSUB.30:
                                ;47957         ;10------------------------------; NOT DONE CHECKING FOR RANGE
                                ;47958         MDR_R[TEMP2],                   ; PUT INTO WORKING TEMPS
                                ;47959         SET OVER(FLAG2),                ; DEFINITE OVERFLOW
U 07AA, 0050,05BE,4030,8467,047B,0 ;47960      PUSH,NEXT/FX.ADJTMPS.LFT2       ; SHIFT LEFT BY 32
                                ;47961
                                ;47962         ;11------------------------------; DONE CHECKING ALL
U 07AB, 0886,85B7,00B0,0047,0000,1 ;47963      M[TEMP8]_0,RETURN [+1]          ; ZERO OUT EXP
```

```
                                ;47964 .TOC    ''       GH FLOAT Subroutines in ROM     : HFLOAT to Integer FX.CVTFI.HSUB.20
                                ;47965
                                ;47966 =00
                                ;47967 FX.CVTFI.HSUB.50:
                                ;47968          ;00-------------------------------; EXP <1-31>
                                ;47969          M[TEMP8]_0,                       ; ZERO OUT EXP
U 07AC, 0486,85B7,0030,0047,007D,A  ;47970      NEXT/FX.CVTFI.HSUB.100            ;
                                ;47971
                                ;47972          ;01-------------------------------; EXP=32 (WB=0)
                                ;47973          MDR_R[TEMP2],                     ; GET INTEGER
                                ;47974          FLAG<2-0>?,                       ; START CHECK FOR OVERFLOW
U 07AD, 0C80,05BE,45B0,8467,0078,2  ;47975      NEXT/FX.CVTFI.CHKOVR              ;
                                ;47976
                                ;47977          ;10-------------------------------; EXP=0 (WB=32)
                                ;47978          M[TEMP8]_0,MDR_0,                 ;
U 07AE, 0086,85B7,00B0,04E7,0000,1  ;47979      RETURN [+1]                       ;
                                ;47980
                                ;47981          ;11-------------------------------; WB >32 EXP >32
                                ;47982          M[TEMP0]_MB+ZLIT0[20],            ;
                                ;47983          DBZ STEP0?,                       ; DONE CHECKING
U 07AF, 0586,0C11,0331,0047,007A,A  ;47984      NEXT/FX.CVTFI.HSUB.30             ;
                                ;47985
                                ;47986 FX.CVTFI.HSUB.100:
                                ;47987          ;-------------------------------;
U 07B0, 0C81,2137,0030,8467,007D,8  ;47988      MDR_(M[MDR] R[TEMP2]).RR.P        ; GET THE INTEGER
                                ;47989
                                ;47990          ;-------------------------------;
                                ;47991          M[TEMP2]_(MB R[TEMP3]).RR.P,      ; GET BITS 0-31
U 07B1, 0C86,2137,0030,C047,007C,9  ;47992      NEXT/FX.LFTSHFT3                  ; GET REMAINING BITS
                                ;47993
                                ;47994 =0
                                ;47995 FX.CVTFI.HSUB.110:
                                ;47996          ;0-------------------------------; DONT CLEAR ROUND BIT
U 07C2, 0880,0036,40B0,0047,0000,1  ;47997      RETURN [+1]                       ;
                                ;47998
                                ;47999          ;1-------------------------------; CLEAR ROUND FOR CVTRHL
                                ;48000          R[TEMP2]_0,                       ;
U 07C3, 0C84,05B7,00B0,8047,0000,1  ;48001      RETURN [+1]                       ;
```

```
                                ;48002 .TOC     ''      GH FLOAT Subroutines in ROM     : HFLOAT to Integer FX.CVTFI.HSUB.20
                                ;48003 =010
                                ;48004 FX.CVTFI.CHKOVR:
                                ;48005          ;010----------                          ; FLAG 2/0=0
                                ;48006          MDR_R[TEMP2],                           ; INTEGER TO MDR
                                ;48007          SET OVER(FLAG2),                        ; OVERFLOW TIME
U 0782, 0050,05BE,4030,8467,007B,2  ;48008      NEXT/FX.CVTFI.CHKOVR.20                 ;
                                ;48009
                                ;48010
                                ;48011          ;011----------                          ; -2**32
                                ;48012          MDR_R[TEMP2],CLEAR FLAG2,               ; INTEGER
                                ;48013          POPTC(FLAG4)?,                          ; TRUNCATE OR ROUND
U 0783, 0010,05BE,4470,8467,007C,4  ;48014      NEXT/FX.CVTFI.CHKOVR.10                 ;
                                ;48015
                                ;48016          ;110----------                          ; PREVIOUS OVERFLOW TIME
                                ;48017          MDR_M[TEMP2],                           ; INTEGER TO MDR
U 0786, 0C80,2002,403D,8467,007B,2  ;48018      NEXT/FX.CVTFI.CHKOVR.20                 ; CONTINUE ON
                                ;48019
                                ;48020          ;111----------                          ; PREVIOUS OVERFLOW TIME
                                ;48021          MDR_M[TEMP2],                           ; INTEGER TO MDR
U 0787, 0C80,2002,403D,8467,007B,2  ;48022      NEXT/FX.CVTFI.CHKOVR.20                 ; CONTINUE ON
                                ;48023
                                ;48024 =0
                                ;48025 FX.CVTFI.CHKOVR.10:
                                ;48026          ;0----------                            ; TRUNCATE
                                ;48027          R[TEMP2]_RB.ASL.1,                      ; PREVIOUSLY 80---0 REMOVING BIT
                                ;48028          WX.NE.0?,                               ; IF -2**32 ONLY
U 07C4, 0484,05F7,0A50,8047,087B,2 331*  ;48029  NEXT/FX.CVTFI.CHKOVR.20                 ;
                                ;48030
                                ;48031          ;1----------                            ; ROUND
                                ;48032          PL [1F],                                ; SET UP TO GET ROUND BIT
U 07C5, 0120,0EF6,4030,F847,007D,C  ;48033      CLEAR POP1C(FLAG4)                      ; GET RID OF SIGNAL
                                ;48034
                                ;48035          ;----------                            ;
                                ;48036          M[TEMP2]_(MB R[TEMP3]).RR.P,            ; GET BITS 0-31
                                ;48037          WX.NE.0?,                               ;
U 07DC, 0886,2137,0A70,C047,007B,2  ;48038      NEXT/FX.CVTFI.CHKOVR.20                 ; CONTINUE TESTING
                                ;48039
                                ;48040 =00
                                ;48041 =01
                                ;48042          ;01----------                           ;
                                ;48043          M[TEMP8]_0,                             ; CLEAR EXP
U 07B1, 0886,85B7,00B0,0047,0000,1  ;48044      RETURN [+1]                             ;
                                ;48045
                                ;48046 FX.CVTFI.CHKOVR.20:
                                ;48047          ;10----------                           ;
                                ;48048          M[TEMP2]_R[TEMP3],                      ; BITS 0-31 TO T2
U 07B2, 0086,25BE,4030,C047,047C,6  ;48049      PUSH,NEXT/FX.ADJTMPS LFT                ; OTHER BITS AND RET-1
                                ;48050
                                ;48051          ;11----------                           ; OVERFLOW TIME
                                ;48052          SET OVER(FLAG2),                        ;
U 07B3, 0450,0036,4030,0047,007B,2  ;48053      NEXT/FX.CVTFI.CHKOVR.20                 ;
```

48053; This page intentionally left blank.

```
                                        ;48054 .TOC "FILLER.MIC"
                                        ;48055 .TOC "REVISION 16.0"
                                        ;48056 ;       Gerard Koeckhoven, CHARLIE MCDOWELL
                                        ;48057
;48058  .NOBIN
;48059
;48060  .TOC "  Revision history"
;48061
;48062  ; REV   EXPLANATION
;48063
;48064  ; 16    Remove OFFA,OFF7 for mem management fix
;48065  ;       crossing page boundary and modifyiing destination
;48066  ; 15    Remove words for CMT061.
;48067  ; 14    Remove words for CMT059
;48068  ; 13    Remove words for CMT058
;48069  ; 12    Remove and add words to filler for CMT057
;48070  ; 11    Remove and add words to filler for CMT056
;48071  ; 10    Delete words for rev 55 changes.
;48072  ; 09    Delete words for rev 54 changes.
;48073  ; 08    Delete words for rev 51 changes.
;48074  ; 07    Delete words for rev 49 changes.
;48075  ; 06    Initial release.
                                        ;48076         .BIN
```

```
                                ;48077 .TOC "  Filler                    : Filler words for unused control store"
                                ;48078
                                ;48079 ;*****************************************************************************
                                ;48080
                                ;48081 .REGION/0,17FF
                                ;48082
                                ;48083          ;-----------------------------------;
U 07F2, 0986,BC37,0035,C047,017F,9      ;48084          M[ERRCOD]_ZLIT0[184.],NEXT/IE.UNUSED.CS         ;07F2
                                ;48085
                                ;48086          ;-----------------------------------;
U 07F3, 0D86,BC37,0036,4047,017F,9      ;48087          M[ERRCOD]_ZLIT0[200.],NEXT/IE.UNUSED.CS         ;07F3
                                ;48088
                                ;48089          ;-----------------------------------;
U 07F4, 0986,BC37,0036,C047,017F,9      ;48090          M[ERRCOD]_ZLIT0[216.],NEXT/IE.UNUSED.CS         ;07F4
                                ;48091
                                ;48092          ;-----------------------------------;
U 07F5, 0D86,BC37,0037,C047,017F,9      ;48093          M[ERRCOD]_ZLIT0[248.],NEXT/IE.UNUSED.CS         ;07F4
                                ;48094
                                ;48095          ;-----------------------------------;
U 0FF7, 0D86,BF37,0031,0047,017F,9      ;48096          M[ERRCOD]_OLIT16[32.],NEXT/IE.UNUSED.CS         ;0FF7
                                ;48097
                                ;48098          ;-----------------------------------;
U 0FFA, 0D86,BF37,0032,0047,017F,9      ;48099          M[ERRCOD]_OLIT16[64.],NEXT/IE.UNUSED.CS         ;0FFA
                                ;48100
                                ;48101          ;       END OF FILLER.MIC
```

```
ALPCTL                  2599 #
        DIVDA           2650 #  14478
        DIVDS           2651 #  14460
        DIVFAST+        2645 #  9552    9589    9594    9599    9606    9612    9617    13079
        DIVFAST-        2647 #  9563    9579    9623    9628    9633    9640    9646    9651
        DIVSLOW+        2646 #
        DIVSLOW-        2648 #
        MULFAST+        2641 #  9168    9173    9178    9183    9189    9194    9198    9202    12849   14207   14218
                        14233   14253   14670   15781
        MULFAST-        2643 #  9207    9212    9217    9222    9228    9233    9237    9241
        MULSLOW+        2642 #
        MULSLOW-        2644 #
        NOP             2600 #  5324    9399    9485    14716   14744   14857   14872   16915   16978   18375   18476
                        18479   18491   18494   18506   18509   18682   22098   37403   37779   37862   38761   38922
                        39152   39879   39913   39969
        REM             2649 #  9453    9464    9470    42118   42124   42133
        WB_ALUF         2636 #  22920   22976
        WB_ALUF.D_S     2638 #
        WB_ALUF.Q_D_S   2639 #
        WB_ALUF.Q_S     2637 #
        WB_LOOPF        2632 #
        WB_LOOPF.D_0    2634 #
        WB_LOOPF.Q_0    2633 #
        WB_LOOPF.Q_D_0  2635 #
        WX_.NOT.S       2625 #  9710    9799    9843    35319   35347   35363   35976   56068   38087   38209   38822
                        41438
        WX_D_.NOT.S     2623 #
        WX_D_DSL.SQL    2627 #
        WX_D_DSL.SQR    2628 #
        WX_D_DSR.SQL    2629 #
        WX_D_DSR.SQR    2630 #
        WX_D_Q.Q_D      2601 #  15379   34999   35008   35325
        WX_D_Q.Q_M      2602 #  15684   35038
        WX_D_Q_.NOT.S   2622 #
        WX_D_Q_S        2618 #  41321   46466   46550   46555
        WX_D_R.Q_D      2603 #
        WX_D_R.Q_M      2604 #  9124    9476    13069
        WX_D_R.Q_XM     2605 #  9300
        WX_D_S          2619 #  5917    5943    5991    7048    9735    11885   12510   12516   12699   12703   12709
                        12715   12721   12724   12729   12734   14086   14116   14739   14835   14840   15123   15351
                        15366   15678   15740   16059   16063   16560   23338   23353   35126   35515   35588   36667
                        38419   42772   42920   43061   43116   46162   46344   46352   46716   46724   46739   47062
                        47493   47498   47503   47508
        WX_D_S.Q_0      2606 #
        WX_D_S.Q_R      2607 #  15519
        WX_D_S.Q_XM     2608 #
        WX_Q.Q_D        2609 #  8350    8435    9335    9669    9725    9730    9834    9839    17595   19278   21010
                        21015   25620   29457   33556   34960   34977   34990   35017   36625   41314   41819   41829
                        41835   42147
        WX_Q.Q_M        2610 #  9665    14052   14062   14073   20103   21762   21852   44430
        WX_Q_.NOT.S     2624 #
        WX_Q_S          2620 #  5559    5563    5582    7729    9603    9637    9693    9754    9786    9808    12056
                        12140   12887   12957   12962   12967   13317   13330   13336   13954   13973   13990   15356
                        15362   15371   15375   16113   16136   19041   19052   19076   19166   27838   27848   27854
                        27858   28176   28725   29380   33207   36673   36847   41170   41761   41770   41781   46299
```

Cross Reference Listing - Field Names and Defined Values

		46394	46419	46444	46478	47345						
WX_R.Q_D	2611 #	5611										
WX_R.Q_M	2612 #	9510	9697	18459	18464	18469	40333	42597	42735	42752	42757	42762
		42881	42900	42905	42910	43041	43046	43051	43139	44348	44352	44386	44401
		44409	44819	44824	44855	44869	44877	44932	44963	44977	44985	45021	45279
		45308	45323	45331	45354	45359	45392	45407	45415	45446	45494	45502	45515
		45821	45852	45866	45874	45898	45903	45959	45967	46008	46012	46057	46065
WX_R.Q_XM	2613 #											
WX_S	2621 #	5327	5352	5357	5362	5365	5384	5402	5413	5431	5471	5476
		5488	5503	5515	5519	5523	5530	5534	5600	5605	5617	5662	5687
		5723	5731	5898	5912	5927	5929	5933	5939	5951	5954	5957	5960
		5963	5966	5973	5977	6127	6132	6142	6148	6155	6159	6163	6166
		6174	6221	6224	6232	6235	6239	6243	6247	6251	6255	6264	6273
		6288	6292	6302	6306	6316	6320	6330	6334	6344	6348	6358	6362
		6372	6376	6386	6390	6400	6404	6414	6418	6432	6482	6486	6502
		6527	6535	6556	6580	6600	6612	6617	6620	6651	6660	6665	6685
		6707	6752	6769	6782	6790	6800	6803	6825	6831	6857	6931	6964
		6972	6977	7013	7021	7036	7094	7113	7160	7169	7173	7176	7234
		7263	7267	7275	7279	7283	7309	7382	7389	7398	7404	7409	7416
		7419	7426	7432	7437	7444	7453	7458	7465	7475	7488	7498	7503
		7510	7520	7525	7532	7542	7547	7556	7568	7577	7588	7594	7638
		7651	7663	7686	7705	7756	7761	7768	7781	7786	7792	7812	7860
		7871	7895	7899	7919	7943	7949	7958	7967	7985	8008	8013	8024
		8050	8093	8163	8167	8173	8186	8190	8194	8205	8496	8510	8521
		8526	8535	9031	9034	9103	9115	9686	9702	9716	9741	9747	9763
		9773	9781	9795	10683	10697	10738	10745	10770	10808	10820	10872	10909
		10914	10925	10929	10933	10976	10981	11029	11034	11092	11106	11114	11121
		11149	11154	11166	11177	11238	11250	11373	11379	11383	11386	11454	11458
		11550	11639	11760	11774	11793	11803	11816	11880	11982	12051	12061	12069
		12127	12135	12145	12154	12165	12389	12410	12541	12545	12565	12569	12590
		12603	12606	12611	12615	12645	12838	12891	12949	12977	12982	12988	13056
		13151	13227	13312	13323	13342	13346	13355	13360	13366	13371	13374	13448
		13688	13692	13706	13715	13761	13780	13861	13949	13959	13968	13977	13981
		13986	14112	14304	14323	14418	14424	14500	14505	14512	14519	14645	14655
		14665	14695	14728	14850	14935	14959	14964	14970	15108	15118	15136	15140
		15146	15156	15161	15169	15187	15197	15212	15215	15220	15225	15231	15242
		15308	15313	15393	15398	15476	15480	15689	15724	15746	15752	15775	15786
		15846	15868	16025	16039	16047	16069	16198	16230	16246	16273	16287	16419
		16435	16477	16486	16555	16578	16588	16633	16649	17004	17048	17117	17122
		17294	17321	17326	17332	17474	17511	17516	17527	17546	17562	17601	18290
		18334	18395	18423	18430	18435	18442	18447	18454	18482	18486	18497	18501
		18512	18516	18659	18663	18698	18702	18707	18714	18723	18729	18734	18738
		18742	18746	18750	18754	18757	18761	19047	19060	19070	20183	20468	20560
		20564	20597	20613	20622	20632	20642	20652	20662	20671	20681	20691	20701
		20711	20720	20965	20972	20975	21115	21136	21259	21910	21968	22428	22435
		22450	22454	22481	22485	22545	22637	22677	22781	22929	22933	22946	22999
		23004	23078	23128	23133	23237	23245	23278	23283	23437	23509	23680	23688
		23922	23932	23946	24098	24422	24462	24471	24477	24494	24525	24540	24675
		24692	24741	24744	24747	24760	24774	24785	24805	24815	24868	24973	24983
		25001	25022	25030	25040	25050	25061	25075	25095	25109	25119	25161	25196
		25210	25276	25288	25299	25367	25634	25648	25713	25790	25796	25931	25935
		25953	25957	25973	25994	26002	26006	26009	26017	26029	26142	26146	26293
		26345	26353	26357	26361	26364	26370	26373	26500	26504	26512	26581	26607
		26819	26895	27013	27017	27022	27026	27043	27051	27055	27086	27101	27140
		27298	27417	27420	27494	27506	27520	27665	27674	27686	27711	27720	27731

Cross Reference Listing - Field Names and Defined Values

27774 27778 27800 27817 27868 27872 27876 27911 27915 27922 27937 27941
27945 27983 28007 28048 28064 28069 28106 28117 28143 28147 28151 28182
28187 28213 28263 28268 28296 28311 28341 28417 28427 28453 28493 28558
28573 28703 28713 28721 28733 28741 28745 28749 28769 28778 28852 28855
29074 29089 29100 29103 29109 29180 29223 29246 29253 29280 29334 29346
29391 29424 29485 29491 29494 29545 29549 29636 29734 29739 29742 29745
29818 29927 29953 29968 29973 29977 29991 29996 30019 30036 30042 30054
30104 30108 30113 30201 30276 30280 30295 30351 30355 30373 30377 30384
30387 30391 30395 30456 30460 30480 30491 30518 30531 30609 30618 30655
30658 30728 30745 30913 30918 30923 30957 30962 30972 30977 31166 31214
31221 31238 31441 31498 31504 31578 31591 31595 31598 31603 31617 31622
31626 31629 31634 31744 31779 31857 31871 31881 31920 31926 32040 32080
32085 32115 32122 32139 32143 32155 32162 32197 32200 32203 32206 32210
32214 32234 32237 32240 32243 32247 32251 32270 32274 32277 32280 32285
32288 32420 32423 32426 32438 32445 32459 32466 32478 32485 32490 32494
32545 32564 32567 32570 32590 32594 32598 32602 32607 32611 32615 32624
32746 32888 32893 32906 32963 32968 32978 33001 33006 33014 33128 33150
33182 33190 33198 33215 33218 33221 33229 33232 33235 33482 33503 33527
33533 33538 33562 33578 33618 33629 33653 33659 33715 33731 34008 34309
34362 34387 34504 34544 34565 34654 34663 34676 34682 34710 34719 34736
34740 34744 35026 35062 35066 35078 35082 35090 35094 35106 35110 35118
35122 35213 35256 35275 35293 35308 35381 35385 35402 35411 35433 35440
35445 35450 35459 35477 35488 35492 35497 35502 35507 35511 35525 35551
35570 35574 35579 35621 35669 35741 35797 35813 35817 35821 35825 35833
35837 35841 35849 35852 35856 35860 35872 35908 35918 35922 35926 35930
35934 35938 35951 35954 35961 35971 35999 36015 36034 36043 36056 36101
36106 36111 36115 36128 36139 36146 36183 36227 36232 36237 36241 36258
36287 36299 36335 36622 36635 36639 37104 37109 37143 37246 37251 37340
37374 37400 37426 37479 37491 37512 37536 37559 37563 37585 37628 37634
37640 37646 37661 37671 37726 37801 37807 37813 37839 37853 37856 37914
37961 37964 37968 37972 37976 38009 38039 38064 38095 38101 38109 38156
38160 38202 38217 38270 38282 38345 38351 38366 38384 38427 38452 38539
38546 38551 38609 38614 38620 38626 38631 38637 38643 38733 38739 38745
38751 38827 38837 38952 39015 39024 39048 39074 39120 39146 39397 39497
39500 39923 39930 40074 40079 40086 40090 40157 40162 40168 40172 40190
40194 40244 40303 40306 40316 40319 40336 40406 40420 40425 40431 40435
40497 40502 40509 40512 40521 40527 40541 40547 40551 40560 40571 40589
40609 40625 40639 40642 40646 40650 40989 40999 41005 41213 41231 41327
41377 41404 41412 41418 41433 41447 41480 41519 41566 41571 41655 41678
41683 41726 41746 41755 41766 42086 42658 44804 45168 45233 45424 45603
45669 45677 46148 46151 46155 46159 46165 46187 46201 46218 46243 46251
46279 46305 46313 46348 46357 46360 46391 46407 46416 46432 46441 46457
46475 46492 46569 46576 46583 46590 46595 46601 46608 46615 46622 46629
46635 46642 46648 46655 46662 46668 46675 46682 46689 46696 46744 46763
46807 46810 46814 46818 46827 46831 46886 46893 46895 46911 46915 46922
46925 46947 46993 46996 47003 47012 47022 47082 47094 47101 47112 47135
47139 47153 47157 47165 47168 47176 47181 47184 47187 47190 47193 47197
47200 47211 47214 47222 47227 47230 47233 47237 47240 47251 47254 47262
47265 47268 47271 47274 47284 47293 47302 47356 47378 47393 47401 47405
47411 47415 47419 47422 47430 47469 47473 47477 47539 47544 47549 47559
47613 47619 47629 47634 47639 47666 47670 47679 47684 47689 47695 47736
47762 47766 47772 47776 47788 47793 47797 47839 47844 47848 47851 47873
47877 47881 47886 47889 47892 47908 47947 47963 47969 47978 47988 47991
48000 48027 48036 48043 48084 48087 48090 48093 48096 48099

WX_S.Q_0 2614 # 7740

Cross Reference Listing - Field Names and Defined Values

	WX_S.Q_R	2615 #	9387	42090								
	WX_S.G_XM	2616 #	11224	21262	21758	21848	27768	28846	29120	32034	32692	
ALU		2654 #										
	A+B+CI	2662 #	5398	5408	5454	5458	5462	5546	5590	5614	5630	5659	5675
		5712	5749	5776	5796	5947	5997	6170	6269	6436	6585	6603	6656
		6701	6779	6787	6795	6817	6903	6910	6917	6922	7041	7065	7077
		7089	7101	7110	7129	7219	7658	7670	7689	7737	7828	7832	7911
		7915	7978	8087	8100	8552	8557	8563	8589	8595	8601	8635	8652
		8708	8919	9043	9055	9058	9489	9770	11326	11340	11344	11351	11633
		11650	11656	11662	11798	12341	12352	12358	12549	12554	12594	12598	12626
		12843	13084	13460	13590	13602	13635	13645	13651	13657	13870	13881	14223
		14228	14238	14241	14247	14258	14441	14683	14710	14825	14976	14991	15126
		15201	15406	15714	15799	15816	15822	15850	15916	16211	16267	16293	16472
		16491	16510	16528	16532	16548	16567	16593	16654	16994	17030	17052	17242
		17303	17491	17550	17613	18240	18244	18405	18692	19096	19148	19187	19248
		19290	19296	19335	19341	19448	19455	19528	19738	20418	20530	20536	20569
		20734	20738	20742	20767	20814	20899	20929	20962	21020	21147	21206	21216
		21272	21438	21449	21453	21467	21471	21531	21542	21546	21588	21612	21615
		21621	21655	21662	21712	21723	21730	21768	21782	21794	21808	21858	21864
		21874	21881	21903	22788	22969	22988	22991	23056	23110	23196	23201	23261
		23324	23329	23449	23453	23473	23514	23530	23535	23550	23825	23833	23838
		23875	23880	23903	23997	24004	24010	24088	24092	24195	24215	24219	24323
		24327	24427	24536	24575	24644	24650	24831	24880	25182	25296	25660	25717
		25806	25868	25872	26255	26283	26337	26341	26593	26645	26876	26899	27048
		27062	27067	27097	27136	27177	27182	27186	27258	27264	27271	27275	27288
		27303	27540	27704	27727	27747	27787	27791	27820	27823	27959	28030	28248
		28257	28333	28337	28506	28510	28562	28582	28588	28601	28660	28665	28773
		28849	28858	28865	29162	29213	29227	29275	29338	29355	29369	29488	29559
		29643	29649	29654	29697	29842	29902	29911	29930	29934	30193	30346	30400
		30451	30488	30521	30595	30630	30635	30713	30755	30759	30873	30878	30967
		31244	31336	31348	31527	31752	32037	32111	32118	32371	32375	32379	32395
		32399	32403	32517	32719	32790	32794	32828	32832	32836	32867	32870	33060
		33064	33119	33496	33518	33893	33948	34197	34318	34354	34358	34368	34405
		34446	34450	34509	34553	34599	34616	34661	34696	35185	35251	35520	35584
		35598	35696	35701	36072	36136	36293	36311	36315	36469	36486	36553	36570
		36678	36683	36687	36696	36700	36745	36781	36808	36853	36865	36871	36878
		36882	36891	37098	37114	37119	37129	37135	37255	37431	37482	37685	37763
		37768	37867	37876	38091	38112	38167	38213	38228	38278	38389	38393	38404
		38433	38437	38443	38472	38561	38577	38653	38698	38705	38711	38716	38721
		38728	38756	38766	38770	38927	38970	39111	39157	39179	39189	39335	39341
		39507	39725	39739	39884	40300	40368	40506	40531	40537	40630	40705	40713
		40766	40774	41024	41035	41042	41068	41485	41489	41493	41499	41504	41599
		41607	41612	41626	41852	41858	41867	41957	41962	42153	42477	42603	42652
		42715	42792	42829	42845	42942	42975	42991	43083	43125	43130	43166	43197
		43207	44377	44396	44424	44454	44492	44507	44522	44527	44533	44544	44567
		44588	44592	44596	44603	44618	44623	44629	44640	44678	44693	44708	44713
		44717	44729	44750	44766	44781	44845	44864	44889	44913	44954	44959	44972
		45001	45031	45033	45083	45088	45101	45136	45147	45199	45220	45245	45300
		45318	45383	45402	45470	45489	45523	45564	45616	45635	45726	45745	45772
		45800	45842	45861	45931	45953	46033	46052	46215	46235	46276	46295	46386
		46402	46427	46452	46487	46780	47740	47744	47749	47856	47859	47862	47865
		47982											
	A+B+CI.BCD	2663 #	7799	7818	26699	26714	26735	26740	26757	26760	26763	27160	27164
		27168	27172	27195	27206	27216	27226	27229	27233	27431	27435	27446	27449
		28461	28522	28524	28527	29511	29514	29519	29529	30229	30505	30708	30741

Cross Reference Listing - Field Names and Defined Values

	30768	31678	31682	31685	31691							
A+B+CI.SL	2665 #	5573	6814	6843	8926	11555	11644	14076	14097	14474	15000	15836
	15841	15922	19524	19582	20224	21330	27813	29499	29899	33115	36307	41257
	41263	41268	41279	41591	41603	41619	41984	42356	46226	46286	46410	46435
	46460	46822	46838	47359	47381							
A+B+CI.SR	2664 #	5586	7725	15805	15855	15860	16581	18400	21281	25608	25686	25858
	25863	26135	26168	26180	26317	26321	26578	26588	26636	27336	27454	27682
	27700	27737	27889	29349	29639	29776	29779	29828	31517	31714	32063	32067
	33492	34547	35143	45056	47396							
A-B-CI	2656 #	5333	5539	5552	5635	5681	5741	5784	5806	6006	6138	6607
	6889	7031	7045	7062	7074	7086	7098	7106	7183	7203	7223	7584
	7614	7626	7634	7675	7733	7795	7814	7907	7923	7973	8075	8079
	8177	8265	8362	8373	8455	8467	8619	8684	8690	8731	8736	8930
	8941	8995	9048	9063	9318	9405	9409	9440	9481	9558	9569	9778
	9805	11010	11129	11135	11139	11180	11464	11807	12221	12236	12579	12619
	12641	12857	13466	13478	13585	13622	13875	14687	15005	15012	15077	15093
	15333	15508	15530	15535	15710	15827	15874	15879	15886	15903	16118	16128
	16140	16482	16524	16666	16722	16733	16761	16791	16797	16802	16833	16844
	17159	17168	17172	17176	18221	18249	18685	18688	19314	19357	19388	19394
	19398	19468	19477	19482	19557	19593	19639	19722	20123	20134	20190	20195
	20200	20444	20475	20503	20508	20513	20819	20913	21040	21045	21102	21152
	21158	21213	21267	21475	21539	21553	21557	21597	21624	21673	21686	21719
	21726	21813	22461	22467	22477	22536	22683	22687	22916	22994	23049	23053
	23060	23064	23085	23106	23115	23119	23144	23186	23190	23253	23265	23269
	23295	23319	23468	23526	23540	23544	23562	23725	23760	23807	23816	23821
	23829	23884	23888	23907	23957	23978	23984	23988	24014	24019	24027	24039
	24047	24061	24102	24107	24115	24189	24238	24250	24313	24342	24365	24485
	24519	24557	24565	24569	24572	24683	24704	24707	24718	24796	24801	24834
	24959	25045	25090	26171	26192	26252	26280	27827	27834	27844	27884	27964
	27992	27995	28017	28027	28075	28078	28154	28245	28284	28322	28327	28347
	28394	28407	28532	28537	28669	28953	28974	29115	29554	29679	29684	29688
	29721	29726	29730	29868	29876	30359	30363	30574	30579	30584	30589	30599
	30624	30640	30785	30829	30840	30883	30887	30892	30908	30926	30987	30997
	31001	31007	31019	31025	31048	31248	31533	31538	31548	31553	31574	31673
	31905	31908	32384	32482	32511	32648	32658	32663	32667	32682	32813	32822
	32941	32947	33018	33575	33609	33614	33637	33648	33902	33956	33959	33992
	34003	34071	34088	34149	34161	34171	34253	34259	34349	34713	34890	34899
	34903	35098	35102	35271	35417	35468	35481	35531	35535	35566	35986	36046
	36051	36254	36303	36331	36341	36357	36517	36630	36715	36817	36841	37330
	37344	37356	37364	37436	37440	37678	37697	37980	37989	38105	38566	38658
	38775	38806	38832	38844	38916	38974	39002	39006	39053	39060	39096	39103
	39117	39166	39184	39209	39268	39274	39759	39805	39813	39833	39948	39959
	40718	40722	40778	40781	41030	41062	41073	41080	41341	41528	41538	41543
	41884	41921	41930	41973	41979	41995	42102	42143	42402	42489	42557	42579
	42776	42782	42788	42925	42931	42938	43066	43072	43079	43190	44367	44483
	44550	44578	44668	44757	44798	44835	45010	45073	45185	45290	45373	45460
	45607	45717	45785	45832	45919	46023	46179	46182	46367	46543	46729	46747
	46907	46951	46989	46999	47069	47336	47369	47780	47808	47935	47942	
A-B-CI.BCD	2657 #	25884	25890	25895	25900	25906	25912	25919	25926	26704	26709	26730
	26745	26808	26811	26814	26827	26835	26838	26841	28457	28469	28484	28488
A-B-CI.SL	2659 #	14455	14470									
A-B-CI.SR	2658 #	9435										
AND	2667 #	5495	5499	5625	5698	5715	5823	6498	6507	7051	7165	7230
	7317	7325	7333	7341	7348	7356	7364	7372	7376	7480	7962	8030
	8057	8062	8067	8072	8143	8198	8239	8244	8897	8903	9269	9274

Cross Reference Listing - Field Names and Defined Values

	9279	9285	9290	9295	9395	9676	9661	11319	11445	11561	11768	11780
	11811	12559	12636	12881	13073	13866	14700	14733	15071	15087	15389	15811
	16290	16622	17608	17634	18207	19082	19086	19173	19177	19236	19241	20175
	20211	20243	20252	20261	20269	20278	20287	20296	20304	20313	20322	20331
	20339	20349	20354	20359	20364	20369	20374	20378	20383	20388	20393	20398
	20403	20408	20465	20539	20572	20607	20617	20626	20636	20646	20656	20666
	20675	20685	20695	20705	20715	20750	20810	20954	21164	21324	21384	21394
	21603	21664	21697	21786	21867	21888	21951	21978	22011	22020	22785	22801
	23030	23034	23038	23095	23167	23171	23175	23204	23332	24225	24231	24256
	24262	24647	24669	24680	24763	24782	24929	24952	25025	25064	25165	25203
	25243	25283	25292	25601	26088	26099	26417	26423	26439	27331	27502	27512
	27516	27525	27529	27533	27537	27796	28476	28499	28597	28617	28792	28796
	29086	29092	29112	29133	29137	29141	29176	29249	29268	29710	29715	29849
	29859	29872	29881	29885	29891	29895	29906	30242	30247	30256	30495	30679
	30731	30771	31012	31061	31227	31230	31241	31411	31421	31435	31587	31649
	32044	32099	32131	32151	32172	32628	32634	32638	32644	32654	32670	32686
	32701	32706	32915	32919	32937	32991	32996	33026	33131	33134	33137	33668
	33672	33676	33819	33831	34033	34145	34167	34301	34393	34440	34458	34498
	34610	34658	34679	34701	34707	35030	35168	35204	35268	35314	35328	35331
	35368	35375	35967	36063	36295	36353	36478	36495	36562	36579	36722	36748
	36755	36904	37194	37282	37311	37316	37321	37693	37718	37736	37743	37830
	37844	38152	38173	38570	39028	39038	39346	39372	39493	40327	41354	41647
	41668	41947	42044	42081	42113	42274	42280	42290	42486	42661	42694	46212
	46230	46874	47281	47573	47651	47757						
AND.SL	2670 #	14660	21142	27498	36142	42492						
AND.SR	2669 #	25655	26641	27356	27360	31522						
ANDNOT	2672 #	6516	6544	6839	7187	7401	7429	7450	7472	7495	7517	7539
	7564	7591	7753	7789	7809	8856	8864	15996	16990	17237	18226	19064
	19116	19120	19128	19153	19207	19211	19219	20127	20138	20428	20453	20778
	20786	20804	21318	21358	25611	26110	26211	26464	29565	29584	29589	29594
	29599	29604	29659	29669	29821	30252	31111	31455	31688	32573	32861	32878
	32883	32903	32952	32983	33037	33051	33465	33720	34036	34185	34285	34297
	34560	34569	34837	34863	35176	35546	36060	36732	36759	36799	37352	37379
	37388	37486	37787	38222	38850	40237	40309	40324	40515	40576	40593	40605
	41718	42175	42307	42673								
B-A-C1	2660 #	7207	8260	8377	8678	8726	9066	9152	9162	9526	9573	9584
	11015	11019	11169	11367	11470	11675	11686	11703	12331	12368	12738	12741
	12744	13061	13615	13673	13677	13682	14429	14772	14782	15083	15098	15102
	15130	15205	15328	15502	15525	15541	16010	17581	18202	18320	18521	18524
	19309	19352	19404	19513	19518	19553	19588	21049	21111	21327	21733	21819
	21823	21906	21919	22748	22972	23234	23487	23766	23811	23943	24073	24082
	24128	24135	24546	25253	27671	28038	28159	28240	29415	29473	29937	30225
	31508	31853	31901	31944	33805	33945	34550	34604	39043	39082	40482	40487
	40492	40621	41100	41106	41111	41118	41174	41179	41184	41191	41372	41464
	41524	41532	41890	41899	41915	41990	42002	42007	42097	42158	42864	43005
	43157	46528	46533	46538	47952							
NOTAND	2673 #	8367	8850	9829	21399	23141	23292	35239	41708	42313	42322	42511
	46983											
OR	2668 #	5370	5375	5388	5392	5417	5421	5438	5441	5448	5451	5507
	5526	5568	5578	5595	5646	5655	5667	5671	5726	5754	5760	5880
	5887	5903	5907	5924	6122	6152	6177	6260	6277	6512	6541	6590
	6647	6678	6743	6746	6748	6756	6760	6764	6772	6806	6811	6828
	6861	6864	6867	6871	6875	6879	6883	6892	6895	6900	6907	6914
	6925	6935	6956	6959	6969	6987	6990	7017	7027	7116	7119	7122
	7125	7180	7190	7192	7198	7215	7226	7238	7271	7287	7290	7294

7386	7392	7396	7413	7423	7441	7447	7462	7469	7485	7492	7507
7514	7529	7536	7560	7581	7655	7667	7694	7698	7721	7749	7804
7823	7853	7876	8097	8103	8121	8127	8131	8135	8139	8146	8150
8153	8278	8291	8306	8320	8332	8356	8387	8403	8415	8420	8440
8459	8471	8492	8501	8506	8606	8640	8695	8780	8806	8812	8819
8824	8869	8909	8913	8936	8946	8985	8992	9018	9022	9027	9038
9071	9098	9147	9157	9305	9309	9314	9323	9327	9339	9376	9381
9415	9424	9445	9494	9498	9501	9505	9514	9518	9522	9706	9720
9760	9791	9812	9816	9820	9825	10679	10692	10702	10732	10751	10763
10773	10804	10829	10991	10995	11001	11101	11110	11145	11157	11233	11243
11258	11334	11358	11440	11478	11485	11570	11629	11667	11696	11756	11787
11824	11888	11896	11907	11913	11928	11933	11977	12046	12160	12227	12232
12336	12347	12378	12395	12404	12524	12531	12575	12585	12693	12833	12863
13052	13089	13156	13165	13232	13244	13453	13472	13571	13576	13581	13595
13609	13618	13629	13640	13663	13698	13766	13771	13785	13796	14038	14043
14048	14058	14069	14082	14093	14102	14107	14212	14309	14314	14328	14339
14434	14450	14493	14508	14639	14649	14676	14705	14720	14748	14757	14762
14767	14777	14789	14794	14802	14807	14814	14819	14866	14940	14947	14952
14984	15113	15151	15175	15182	15191	15238	15299	15304	15318	15340	15417
15421	15425	15471	15486	15493	15693	15698	15706	15729	15736	15770	15792
15892	15909	15985	15991	16005	16013	16029	16052	16073	16124	16203	16225
16234	16238	16242	16255	16259	16262	16277	16284	16414	16425	16428	16440
16447	16452	16456	16460	16466	16496	16501	16505	16515	16520	16538	16543
16628	16637	16910	16922	16926	16942	16973	16986	17025	17070	17075	17110
17155	17164	17195	17199	17210	17214	17218	17221	17232	17274	17278	17311
17316	17441	17448	17454	17460	17466	17470	17477	17481	17486	17499	17507
17534	17578	17587	17592	17616	17619	17625	17629	17638	18198	18217	18230
18235	18274	18278	18286	18315	18330	18363	18367	18371	18378	18382	18427
18439	18451	18651	18656	18671	18678	18711	18718	18726	18765	18769	18772
18914	18970	18975	18980	18985	18990	18995	19000	19004	19009	19021	19108
19112	19124	19142	19163	19199	19203	19215	19283	19302	19321	19329	19348
19381	19409	19413	19417	19444	19460	19464	19473	19492	19533	19539	19545
19564	19571	19575	19578	19607	19611	19615	19619	19623	19627	19631	19635
19643	19658	19663	19668	19673	19678	19683	19688	19692	19698	19701	19715
19719	19726	19733	20098	20110	20116	20144	20160	20164	20168	20186	20197
20203	20207	20215	20219	20229	20233	20239	20248	20257	20265	20274	20283
20292	20300	20309	20318	20327	20335	20344	20414	20424	20433	20437	20441
20458	20462	20471	20479	20486	20490	20494	20498	20518	20556	20579	20584
20589	20594	20603	20724	20730	20745	20756	20764	20773	20791	20796	21005
21025	21030	21035	21099	21106	21122	21126	21131	21201	21221	21228	21232
21239	21243	21248	21253	21275	21310	21361	21389	21434	21442	21458	21464
21482	21512	21517	21521	21535	21549	21584	21600	21607	21618	21650	21668
21678	21681	21693	21701	21705	21715	21771	21775	21780	21790	21801	21804
21870	21878	21892	21896	21899	21916	21972	21988	21994	21997	22002	22008
22015	22046	22054	22057	22092	22418	22424	22446	22457	22473	22496	22502
22508	22514	22525	22531	22541	22556	22561	22567	22574	22581	22596	22603
22609	22615	22623	22627	22630	22642	22652	22661	22666	22673	22697	22704
22710	22715	22721	22725	22730	22745	22751	22761	22764	22772	22804	22939
22949	22952	22955	22985	23007	23017	23021	23045	23073	23090	23102	23137
23154	23158	23182	23210	23214	23225	23230	23248	23258	23288	23306	23310
23342	23357	23464	23479	23483	23493	23497	23500	23520	23555	23569	23573
23669	23674	23684	23693	23699	23702	23730	23734	23782	23786	23844	23848
23897	23928	23936	23969	23974	23992	24001	24024	24078	24112	24245	24307
24317	24333	24371	24376	24433	24437	24442	24446	24450	24456	24489	24500
24506	24515	24531	24551	24560	24629	24632	24635	24640	24655	24664	24672

24678	24688	24697	24701	24712	24715	24721	24735	24752	24757	24768	24778
24789	24792	24808	24819	24824	24828	24843	24847	24854	24859	24863	24873
24876	24883	24925	24933	24936	24944	24948	24956	24963	24969	24979	24988
24992	24997	25015	25035	25056	25069	25080	25084	25098	25102	25113	25122
25128	25168	25177	25207	25213	25249	25262	25302	25307	25312	25318	25322
25325	25357	25363	25371	25376	25379	25382	25388	25394	25401	25405	25625
25629	25639	25643	25676	25680	25691	25800	25817	25821	25827	25832	25837
25842	26095	26104	26123	26125	26130	26150	26153	26176	26184	26189	26198
26216	26223	26237	26240	26274	26277	26302	26306	26333	26348	26428	26435
26444	26451	26472	26475	26480	26483	26486	26489	26496	26509	26515	26603
26615	26620	26623	26626	26670	26674	26678	26682	26789	26793	26847	26852
27007	27029	27032	27035	27038	27071	27075	27112	27117	27121	27125	27129
27145	27292	27315	27325	27328	27341	27374	27378	27423	27427	27441	27465
27668	27677	27692	27696	27723	27752	27919	27949	27970	27979	27989	28003
28014	28021	28060	28086	28090	28109	28121	28125	28128	28131	28134	28166
28171	28208	28217	28278	28302	28306	28397	28401	28411	28554	28567	28593
28606	28622	28632	28636	28640	28686	28691	28717	28729	28737	28764	28783
28809	28823	28827	28831	28835	28942	28945	28956	28962	28971	28978	28982
29083	29097	29106	29124	29129	29145	29154	29159	29166	29184	29201	29218
29234	29238	29257	29272	29284	29360	29365	29376	29402	29409	29412	29421
29427	29462	29469	29478	29481	29524	29534	29538	29542	29570	29574	29578
29615	29623	29627	29663	29674	29692	29702	29706	29748	29754	29758	29768
29773	29808	29814	29831	29834	29854	29919	29924	29949	29981	29986	30001
30005	30009	30011	30025	30033	30039	30045	30049	30083	30089	30094	30099
30118	30125	30130	30174	30180	30185	30189	30205	30210	30215	30220	30265
30271	30284	30289	30324	30331	30336	30342	30369	30427	30433	30438	30443
30447	30454	30468	30484	30509	30535	30567	30606	30644	30648	30664	30668
30673	30723	30826	30832	30836	30848	30852	30856	30860	30864	30896	30902
30936	30941	30946	30982	30990	31032	31039	31043	31068	31108	31119	31123
31120	31128	31133	31139	31163	31172	31175	31178	31181	31218	31224	31233
31235	31252	31256	31260	31285	31290	31299	31305	31312	31333	31415	31426
31430	31460	31469	31479	31482	31493	31703	31707	31719	31775	31860	31874
31884	31887	31891	31911	31914	31923	31929	31932	31938	31941	31947	32029
32049	32054	32058	32071	32089	32093	32096	32135	32158	32176	32442	32463
32498	32501	32514	32525	32535	32550	32554	32586	32723	32727	32730	32733
32737	32843	32847	32857	32925	32929	33011	33030	33040	33046	33073	33077
33082	33146	33155	33164	33167	33171	33175	33473	33487	33500	33506	33509
33514	33522	33525	33541	33545	33549	33553	33571	33623	33632	33642	33663
33726	33737	33776	33782	33788	33792	33798	33809	33813	33835	33858	33862
33898	33907	33916	33952	33998	34011	34020	34028	34067	34075	34084	34092
34132	34153	34157	34177	34192	34240	34263	34267	34273	34279	34290	34294
34304	34314	34372	34408	34432	34435	34462	34489	34494	34501	34535	34542
34556	34594	34607	34613	34619	34623	34666	34669	34673	34685	34688	34692
34724	34728	34732	34754	34758	34841	34845	34849	34867	34872	34876	34886
34952	34956	34964	34967	34974	34982	35034	35042	35046	35058	35070	35074
35086	35130	35135	35139	35147	35151	35159	35163	35172	35179	35190	35195
35199	35208	35217	35221	35230	35234	35243	35247	35260	35265	35279	35284
35287	35298	35303	35334	35337	35342	35353	35356	35359	35372	35390	35393
35398	35422	35427	35430	35464	35473	35539	35557	35593	35607	35612	35616
35624	35629	35635	35639	35644	35655	35746	35759	35763	35770	35774	35786
35790	35794	35805	35829	35845	35868	35876	35880	35884	35888	35892	35900
35943	35957	35982	35991	35996	36004	36012	36018	36026	36038	36065	36076
36125	36131	36151	36156	36160	36166	36173	36263	36267	36271	36275	36279
36283	36327	36348	36360	36465	36474	36482	36491	36499	36503	36508	36522
36527	36549	36558	36566	36575	36583	36587	36643	36705	36718	36729	36736

	36739	36763	36767	36771	36776	36785	36791	36793	36802	36812	36857	36861
	36875	36885	36888	36899	36908	36911	36914	36916	36919	36922	36929	36935
	36941	36965	37199	37204	37226	37230	37235	37240	37265	37270	37274	37278
	37287	37294	37298	37302	37306	37326	37360	37368	37384	37396	37413	37419
	37422	37509	37651	37657	37666	37704	37722	37733	37747	37755	37773	37776
	37810	37816	37819	37872	37908	37911	37984	37993	37997	38001	38005	38163
	38225	38262	38267	38275	38289	38292	38296	38301	38354	38357	38370	38373
	38379	38399	38408	38456	38460	38466	38476	38482	38486	38492	38662	38855
	39011	39066	39070	39078	39086	39090	39162	39215	39222	39329	39338	39354
	39364	39381	39385	39403	39407	39411	39415	39419	39427	39430	39434	39441
	39445	39449	39452	39462	39474	39478	39644	39692	39722	39729	39736	39743
	39755	39801	39809	39816	39874	39889	39895	39899	39902	39908	39918	39935
	39938	39964	39974	39979	39983	39987	39990	39996	40033	40039	40049	40054
	40059	40065	40070	40095	40140	40149	40153	40179	40184	40199	40207	40212
	40217	40221	40225	40232	40241	40249	40255	40260	40265	40297	40313	40321
	40330	40365	40375	40380	40390	40393	40398	40439	40441	40444	40555	40566
	40584	40597	40636	40655	40692	40701	40755	40762	40814	40821	40854	40862
	40869	40873	40877	40881	40885	40901	40905	40909	40913	40929	40933	40937
	40941	40994	41021	41059	41097	41135	41138	41143	41148	41154	41206	41221
	41225	41247	41251	41276	41297	41301	41305	41332	41346	41366	41400	41408
	41423	41451	41460	41587	41651	41672	41712	41722	41742	41775	41785	41801
	41805	41813	41824	41863	41895	41926	41952	41969	42014	42019	42026	42032
	42037	42048	42052	42056	42060	42076	42107	42163	42191	42195	42199	42241
	42247	42252	42257	42285	42318	42375	42379	42383	42388	42393	42397	42411
	42426	42434	42438	42442	42447	42464	42515	42530	42537	42541	42545	42550
	42565	42570	42611	42619	42624	42629	42643	42655	42666	42706	42719	42739
	42743	42748	42767	42797	42802	42808	42814	42820	42838	42854	42859	42886
	42891	42896	42916	42947	42952	42958	42966	42984	43000	43022	43027	43032
	43037	43057	43087	43091	43097	43103	43108	43147	43152	43177	43181	43185
	43202	43213	43225	43230	43235	44357	44361	44372	44382	44391	44435	44442
	44448	44472	44476	44488	44496	44500	44504	44511	44571	44583	44600	44607
	44657	44661	44673	44682	44686	44690	44697	44746	44762	44770	44774	44778
	44785	44840	44850	44859	44895	44902	44908	44937	44944	44949	44968	44991
	44996	45016	45025	45061	45066	45078	45092	45096	45106	45114	45120	45125
	45172	45192	45205	45210	45215	45225	45251	45274	45295	45305	45313	45336
	45364	45368	45378	45388	45397	45442	45451	45455	45465	45475	45479	45484
	45529	45533	45539	45547	45553	45558	45589	45593	45598	45611	45621	45625
	45630	45640	45648	45657	45673	45682	45702	45706	45721	45731	45735	45740
	45750	45758	45780	45794	45817	45837	45847	45856	45908	45913	45925	45936
	45942	45947	45977	45982	45987	46028	46038	46042	46047	46076	46084	46170
	46175	46364	46372	46377	46382	46472	46495	46521	46560	46563	46566	46598
	46632	46665	46720	46771	46775	46869	46878	46980	47018	47052	47059	47066
	47204	47244	47326	47331	47341	47352	47646	47654	47719	47731	47814	47828
	47832	47835	47958	47973	48006	48012	48017	48021	48048			
XOR	2671 #	5467	5693	5892	6228	6280	6284	6298	6312	6326	6340	6354
	6368	6382	6396	6410	6424	6463	6470	6477	6822	6886	6951	7054
	7058	7070	7082	7313	7321	7329	7337	7344	7352	7360	7368	7643
	7647	7713	7717	7745	7849	8296	8300	8311	8315	8762	8768	8775
	10816	10825	10867	10878	12033	12040	12371	13670	17521	17539	21958	30498
	30689	30695	30699	30734	30780	35436	35455	36030	39375	41457	42209	42215
	42220	46238	46246	46301	46308	46398	46423	46448	46482	46572	46579	46586
	46593	46604	46611	46618	46625	46638	46645	46651	46658	46671	46678	46685
	46692	46735	46882	46889	46918	47008	47015	47090	47097	47108	47161	47172
	47207	47218	47247	47258	47298	47375						
ALUC1	2675 #											

		45096	45101	45192	45199	45215	45220	45295	45300	45313	45318	45378	45383
		45397	45402	45465	45470	45484	45489	45611	45616	45630	45635	45721	45726
		45740	45745	45817	45837	45842	45856	45861	45925	45931	45947	45953	46028
		46033	46047	46052									
	ZERO	2704 #	8087	8320	8387	9706	11224	14649	14947	19524	19571	19578	20207
		20219	20233	22496	22556	22596	22661	22710	23449	23453	23766	23978	23988
		24039	24082	24128	24195	24225	24231	24256	24262	24313	24342	24752	24768
		24824	24936	25035	25069	25318	25382	26302	27727	28849	28865	29154	29162
		29184	29227	29234	29257	29275	29284	29338	29369	29421	29427	29554	29570
		29574	32037	32111	32118	32442	32463	32498	33992	38167	41251	41276	41591
		41603	41619	42438	42655	42719	42767	42916	43057	43225	43230	43235	46882
		46889	46989	46999	47008	47018							
BUS		2708 #											
	GRANT	2739 #	38146										
	IOINIT	2736 #	5323	5328	5334	5947	35562	35565					
	NOP	2709 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
		9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
		9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
		14235	14255	14673	15783								
	PRB.RD	2729 #	20552	21209	21218	35585	35599	35612	35625	35632	35703	36125	36470
		36488	36679	37810	37816	38418	40028	40042	40250	40365	40371	40383	40441
		40567	40577	40585	40594	40620	40629	40654	40657	47193	47204	47233	47244
	PRB.RD.MODE	2730 #	40198	40554									
	PRB.RD.PTE	2731 #	35696	35698	40075	40080	40421	40426					
	PRB.RD.PTE.K	2732 #	5886	5911	5918	6005	35131	36964	37798	38887	38901	40094	40183
		40206	40643										
	PRB.WR	2733 #	20154	21105	21593	21674	21689	21797	21884	29207	29619	36554	36572
		39796	40687	40751	40810								
	PRB.WR.MODE	2734 #	40852										
	PRB.WR.PTE	2735 #	40158	40163									
	PRINIT	2737 #	5940										
	READ	2710 #	5545	5621	5686	5730	5753	5789	5799	5817	6778	6786	6794
		15718	15758	16019	16033	16217	16609	16616	16641	17009	17013	17018	17039
		17081	17089	17260	17265	17284	17555	19537	19543	19737	20109	20115	20535
		20580	20585	20590	20611	20620	20630	20640	20650	20660	20669	20679	20689
		20699	20709	20718	20760	21272	21446	21527	21530	21659	21779	21861	21877
		23756	23803	24201	24205	24480	26345	29397	29841	29910	30117	30214	30450
		32407	32411	32434	32455	32474	32489	33499	33513	34378	36870	36881	39334
		39338	42602	42651	42714	42819	42828	42844	43165	43214	43242	44415	44423
		44436	44443	44453	44538	44543	44634	44639	44723	44728	44903	44912	45026
		45037	45130	45135	45239	45244	45337	45419	45508	45521	45538	45546	45552
		45563	45652	45663	45789	45799	45972	46083	46767	47150	47196	47236	47345
		47365											
	READ.LNG	2712 #	18409	18413	18417	18427	18439	18451	19040				
	READ.LNG.MOD	2713 #	18695	18715	19051								
	READ.MOD	2711 #	42965	42974	42990	43251	44883	44888	44896	45764	45771	45779	45880
		46069	46075										
	READ.MOD.LCK	2715 #	8644	19075	21965	22049	31072						
	READ.NT	2714 #	39649	40312	40518	40580	40600						
	READ.PHY	2716 #	36139	36324	36687	36696	36711	36719	36735	36738	36758	36762	36766
		36788	36817	37211	37217	38286	40633						
	READ.SEC	2717 #	39652										
	RETCMK	2738 #	39364										
	WRITE	2719 #	5563	5568	5578	5595	5630	5671	6900	6907	6914	7226	8291
		8362	8373	8420	8459	8471	8501	8557	8606	8695	8731	8780	8824

Cross Reference Listing - Field Names and Defined Values

		8869	9501	14767	14772	16931	19291	19297	19389	19643	19726	20197	20229
		20239	20248	20257	20265	20274	20283	20292	20300	20309	20318	20327	20335
		20344	20424	20433	20441	20453	20468	21147	21457	21464	21482	21538	21611
		21618	21624	21705	21711	21726	21733	21812	21906	21919	23045	23102	23182
		23258	23306	23310	23473	23520	25713	26895	28171	28296	28593	28617	28622
		28632	28809	28823	28827	28831	28835	29692	29702	29706	30045	30125	30130
		30265	30271	30284	30289	30355	30373	30377	30387	30531	30535	30609	30618
		30708	30745	30852	30856	30860	30864	30923	30957	30962	30972	30977	31744
		32843	32847	32925	32929	32964	32969	33011	33030	33040	33952	34435	34494
		36360	36776	36812	37352	37360	37368	37993	37997	38001	38005	38009	38039
		38482	38486	38570	38662	41191	41213	41499	41538	41852	41884	42241	42307
		42375	42383	42393	42537	42545	42550	42565	42570	46170	46175	46870	46878
		47052	47059	47144	47331	47359	47381						
	WRITE.LNG	2721 #	18711	18722	18733	19108	19112	19116	19120				
	WRITE.NOREG	2720 #	8350	8357	8403	8415	8435	8441	8521	8526	8535	8596	8620
		8685	8709	8769	8813	8857	8914	8931	8942	8986	9044	9049	9072
		9319	9328	9336	9340	9514	9518	9522	9526	9687	9721	9725	9730
		9736	9742	9749	9754	9764	9774	9782	9834	9839	10693	10698	10703
		10753	10765	10775	10821	10830	11259	11479	11486	11668	11676	11687	11697
		11704	11776	11789	11897	11908	11914	11929	11934	13167	13246	13772	13797
		14315	14340	14721	14749	14803	14808	14815	14820	15319	15487	15494	16741
		16754	16769	16781	16810	16825	16866	16875	16893	16949	16955	18241	18245
		18278	18291	19449	21036	21041	21361	21550	21805	21900	30000	40989	40999
		41005	41025	41036	41043	41063	41074	41081	41101	41112	41119	41258	41269
		41280	41566	41571	41627	41656	41762	41771	41820	41830	41836	41859	41891
		42210	42221	42620	42630								
	WRITE.NT	2722 #	39673	39683	39817								
	WRITE.NT.LNG	2723 #	40216	40231									
	WRITE.PHY	2725 #	36857	36875	36885	36899	36911	36929	36935	36941	40178		
	WRITE.SEC	2726 #	39687	39730	39744								
	WRITE.UL	2724 #	8652	19124	19128	21997	22002	22015	22057	22092	26211		
	WRITE.UL.SEC	2727 #	39677										
BUT		2742 #											
	BCDCHK	2808 #	7800	7819	30229	30506	30710	30742	30768				
	BR.SC-4.INT-TS	2820 #	30844	30874	30931	31057	32385	32517	32814	32823	35399	36020	39501
	BRA.ON.ADD	2787 #	8498	8603	8692	8777	8821	8866	11252	18368	18396	19143	19279
		19514	39223										
	BUTXB	2792 #	38414	39997									
	CCBR	2811 #	5328	5328	5335	5335	5358	5358	5366	5366	5385	5385	5399
		5399	5403	5403	5409	5409	5455	5455	5459	5459	5463	5463	5472
		5472	5477	5477	5488	5488	5496	5496	5500	5500	5504	5504	5508
		5508	5515	5515	5519	5519	5523	5523	5531	5531	5535	5535	5540
		5540	5560	5560	5583	5583	5591	5591	5601	5601	5605	5605	5614
		5614	5617	5617	5626	5626	5630	5630	5659	5659	5662	5662	5676
		5676	5682	5682	5687	5687	5699	5699	5712	5712	5716	5716	5723
		5723	5733	5733	5750	5750	5785	5785	5796	5796	5824	5824	5899
		5899	5913	5913	5920	5920	5934	5934	5940	5940	5943	5943	5951
		5951	5954	5954	5957	5957	5963	5963	5968	5968	5978	5978	5992
		5992	5998	5998	6007	6007	6128	6128	6132	6132	6142	6142	6149
		6149	6156	6156	6160	6160	6171	6171	6174	6174	6221	6221	6225
		6225	6229	6229	6233	6233	6236	6236	6240	6240	6244	6244	6248
		6248	6256	6256	6265	6265	6270	6270	6274	6274	6281	6281	6285
		6285	6289	6289	6293	6293	6299	6299	6303	6303	6307	6307	6313
		6313	6317	6317	6321	6321	6327	6327	6331	6331	6335	6335	6341
		6341	6345	6345	6349	6349	6355	6355	6359	6359	6363	6363	6369

6369	6373	6373	6377	6377	6383	6383	6387	6387	6391	6391	6397
6397	6401	6401	6405	6405	6411	6411	6415	6415	6419	6419	6425
6425	6433	6433	6464	6464	6471	6471	6478	6478	6487	6487	6503
6503	6508	6508	6528	6528	6536	6536	6542	6542	6545	6545	6557
6557	6613	6613	6662	6662	6667	6667	6695	6709	6709	6753	6753
6803	6803	6825	6825	6832	6832	6858	6858	6932	6932	6965	6965
6978	6978	7014	7014	7032	7033	7037	7037	7042	7042	7048	7048
7051	7051	7055	7055	7059	7059	7062	7062	7066	7066	7071	7071
7074	7074	7078	7078	7083	7083	7086	7086	7090	7090	7095	7095
7098	7098	7102	7102	7107	7107	7107	7110	7110	7166	7166	7170
7170	7173	7173	7177	7177	7184	7184	7187	7187	7231	7231	7235
7235	7264	7264	7268	7268	7276	7276	7280	7280	7284	7284	7310
7310	7314	7314	7322	7322	7330	7330	7338	7338	7345	7345	7353
7353	7361	7361	7369	7369	7373	7373	7378	7378	7383	7383	7405
7405	7410	7410	7420	7420	7433	7433	7438	7438	7454	7454	7459
7459	7466	7466	7476	7476	7482	7482	7499	7499	7504	7504	7521
7521	7526	7526	7533	7533	7543	7543	7548	7548	7557	7557	7561
7561	7565	7565	7569	7569	7578	7578	7581	7581	7591	7591	7595
7595	7615	7615	7627	7627	7635	7635	7644	7644	7648	7648	7652
7652	7664	7664	7676	7676	7690	7690	7706	7706	7714	7714	7718
7718	7729	7729	7741	7741	7746	7746	7750	7750	7753	7753	7757
7757	7762	7762	7769	7769	7789	7789	7809	7809	7829	7829	7833
7833	7850	7850	7861	7861	7873	7873	7908	7908	7912	7912	7915
7915	7944	7944	7949	7949	7958	7958	7963	7963	7975	7975	7986
7986	8009	8009	8014	8014	8026	8026	8032	8032	8058	8058	8063
8063	8068	8068	8072	8072	8076	8076	8080	8080	8094	8094	8143
8143	8187	8187	8191	8191	8199	8199	8206	8206	8910	8910	8915
8927	8950	8989	9027	9027	9039	9039	9104	9301	9310	9314	9315
9324	9324	9388	9396	9396	9401	9401	9416	9416	9420	9420	9425
9425	9430	9430	9441	9441	9449	9449	9454	9459	9459	9465	9471
9559	9559	9570	9570	9600	9634	9670	9670	9690	9690	9694	9694
9770	9770	9778	9778	9830	10817	10817	10825	10825	10868	10868	10879
10879	10926	10926	10930	10930	10934	10934	11029	11029	11034	11034	11149
11149	11154	11154	11329	11329	11336	11337	11340	11340	11352	11353	11499
11712	11771	11771	11782	11782	11799	11799	11813	11813	11885	11885	12035
12035	12042	12042	12222	12224	12238	12238	12333	12333	12343	12343	12372
12372	12385	12385	12400	12400	12513	12513	12519	12519	12550	12550	12556
12556	12562	12562	12580	12580	12594	12594	12621	12621	12633	12637	12637
12642	12642	12700	12700	12706	12706	12712	12712	12716	12716	12721	12721
12726	12726	12731	12731	12734	12734	12858	12858	12878	12878	13085	13085
13467	13469	13480	13480	13588	13588	13625	13625	13670	13670	13684	13684
13702	13702	13878	13878	14088	14088	14113	14113	14118	14118	14426	14426
14442	14442	14446	14446	14489	14501	14501	14505	14505	14666	14666	14689
14690	14690	14702	14702	14713	14713	14735	14735	14837	14837	14843	14843
14973	14973	15007	15008	15008	15015	15016	15016	15073	15073	15079	15079
15084	15084	15088	15088	15095	15095	15098	15098	15123	15123	15166	15166
15329	15330	15334	15335	15386	15390	15390	15418	15418	15422	15422	15426
15426	15515	15690	15690	15711	15711	15742	15742	15777	15777	15812	15812
15830	15830	15847	15847	15876	15876	15880	15880	15888	15889	15905	15905
15997	15997	16060	16060	16064	16064	16069	16069	16114	16114	16119	16120
16120	16137	16137	16141	16141	16200	16200	16230	16230	16269	16290	16290
16435	16435	16483	16483	16511	16525	16525	16550	16563	16563	16574	16574
16599	16599	16604	16604	16624	16624	16668	16668	16991	16991	16996	16997
17118	17118	17123	17123	17160	17160	17169	17169	17173	17173	17177	17177
17238	17238	17245	17246	17474	17474	17562	17562	17566	17566	17592	17592

Cross Reference Listing - Field Names and Defined Values

17610	17610	17634	17634	18866	18871	18876	18881	18886	18891	18896	18901
19042	19042	19053	19053	19065	19065	19077	19077	19154	19154	19167	19167
19237	19237	19242	19242	19303	19304	19310	19311	19315	19316	19349	19349
19353	19354	19358	19359	19400	19405	19470	19470	19479	19479	19484	19484
19529	19583	19583	20124	20124	20129	20129	20135	20135	20140	20140	20156
20156	20156	20177	20177	20200	20200	20211	20211	20244	20244	20253	20253
20262	20262	20270	20270	20279	20279	20288	20288	20297	20297	20305	20305
20314	20314	20323	20323	20332	20332	20340	20340	20350	20350	20355	20355
20360	20360	20365	20365	20370	20370	20375	20375	20379	20379	20384	20384
20389	20389	20394	20394	20399	20399	20404	20404	20409	20409	20429	20429
20438	20438	20450	20450	20455	20455	20465	20465	20469	20469	20491	20491
20495	20495	20499	20499	20536	20536	20540	20540	20545	20545	20548	20548
20553	20553	20573	20573	20608	20608	20614	20614	20618	20618	20623	20623
20627	20627	20633	20633	20637	20637	20643	20643	20647	20647	20653	20653
20657	20657	20663	20663	20667	20667	20672	20672	20676	20676	20682	20682
20686	20686	20692	20692	20696	20696	20702	20702	20706	20706	20712	20712
20716	20716	20721	20721	20742	20742	20751	20751	20780	20780	20788	20788
20792	20792	20806	20806	20811	20811	20901	20901	20914	20914	20930	20930
20950	20955	20955	20962	20962	20966	20966	20972	20972	20976	20976	21031
21032	21046	21046	21050	21051	21102	21102	21117	21117	21137	21137	21143
21144	21166	21166	21206	21206	21213	21213	21263	21263	21320	21320	21324
21324	21327	21327	21350	21386	21386	21396	21396	21412	21439	21439	21454
21454	21468	21468	21472	21472	21543	21543	21546	21546	21604	21604	21615
21615	21621	21621	21655	21655	21665	21665	21698	21698	21730	21730	21758
21758	21787	21787	21794	21794	21808	21808	21848	21848	21858	21858	21868
21868	21874	21874	21889	21889	21903	21903	21912	21912	21952	21952	21969
21969	21979	21979	21999	21999	22004	22004	22012	22012	22021	22021	22429
22429	22482	22482	22486	22486	22547	22547	22638	22638	22677	22677	22785
22785	23001	23001	23004	23004	23031	23031	23035	23035	23039	23039	23050
23050	23079	23079	23096	23096	23107	23107	23129	23129	23168	23168	23172
23172	23176	23176	23187	23187	23191	23205	23205	23237	23237	23245	23245
23279	23279	23320	23333	23333	23437	23437	23469	23469	23479	23688	23688
23694	23694	23761	23761	23808	23808	23818	23818	23922	23922	23947	23947
23957	23957	23979	23989	24015	24015	24040	24062	24062	24083	24103	24103
24129	24251	24251	24314	24314	24343	24343	24367	24367	24423	24423	24463
24463	24472	24472	24477	24477	24485	24486	24495	24495	24526	24526	24541
24541	24647	24647	24665	24665	24669	24669	24680	24680	24684	24684	24708
24708	24760	24760	24782	24782	24797	24797	24802	24802	24805	24805	24835
24835	24869	24869	24930	24930	24933	24933	24952	24952	24959	24959	25025
25025	25041	25041	25047	25047	25051	25051	25092	25092	25110	25110	25119
25119	25165	25165	25204	25204	25243	25243	25284	25284	25292	25292	25299
25299	25368	25368	25612	25612	25634	25634	25650	25650	25655	25677	25677
25681	25681	25714	25714	25792	25792	25796	25797	25797	25843	25848	25851
25931	25936	25973	25973	25994	25994	26003	26003	26006	26006	26009	26009
26017	26017	26029	26029	26111	26111	26143	26143	26146	26146	26172	26172
26212	26212	26293	26293	26353	26357	26361	26361	26364	26364	26370	26370
26373	26373	26465	26465	26501	26501	26504	26504	26583	26583	26603	26641
26767	26790	26790	26794	26794	26819	26848	26848	26853	26853	26896	26896
27007	27014	27014	27017	27017	27023	27023	27026	27026	27044	27044	27051
27051	27055	27063	27063	27067	27067	27102	27102	27254	27333	27333	27375
27375	27379	27379	27665	27665	27687	27687	27732	27732	27775	27775	27779
27779	27797	27797	27801	27801	27817	27817	27827	27827	27835	27835	27839
27839	27845	27845	27849	27849	27855	27855	27859	27859	27868	27868	27872
27872	27876	27876	27911	27911	27916	27916	27984	27984	28008	28008	28027
28027	28049	28049	28065	28065	28075	28075	28106	28106	28117	28117	28134

Cross Reference Listing - Field Names and Defined Values

28144	28144	28147	28147	28155	28155	28178	28178	28183	28183	28188	28188
28213	28213	28245	28245	28248	28249	28264	28264	28268	28268	28279	28279
28285	28285	28298	28298	28312	28312	28323	28324	28328	28328	28328	28334
28334	28338	28338	28341	28341	28418	28418	28428	28428	28453	28453	28477
28477	28494	28494	28500	28500	28506	28506	28511	28534	28534	28538	28538
28563	28563	28583	28584	28589	28589	28598	28598	28601	28601	28619	28619
28661	28661	28665	28665	28670	28670	28770	28770	28774	28774	28779	28779
28783	28783	28788	28793	28793	28797	28797	28824	28824	28828	28828	28836
28836	28855	28855	29074	29074	29089	29089	29112	29112	29116	29116	29134
29134	29138	29138	29142	29142	29176	29176	29246	29246	29250	29250	29269
29269	29381	29381	29488	29488	29491	29491	29511	29511	29535	29535	29538
29538	29560	29560	29567	29567	29585	29585	29590	29590	29595	29595	29600
29600	29605	29605	29636	29636	29660	29660	29670	29670	29680	29680	29684
29684	29689	29689	29698	29698	29712	29712	29717	29717	29723	29723	29727
29727	29731	29731	29818	29818	29822	29822	29828	29828	29836	29836	29843
29843	29850	29850	29869	29869	29869	29877	29877	29912	29912	29927	29986
30029	30105	30105	30126	30131	30137	30141	30202	30202	30226	30226	30243
30244	30244	30249	30249	30253	30253	30257	30257	30257	30267	30267	30273
30273	30277	30277	30281	30281	30286	30286	30296	30296	30352	30352	30370
30374	30374	30380	30380	30388	30388	30400	30451	30451	30457	30457	30461
30461	30481	30481	30492	30492	30499	30499	30518	30518	30532	30532	30575
30575	30580	30581	30585	30586	30586	30601	30601	30611	30611	30620	30620
30625	30626	30626	30632	30632	30636	30636	30641	30641	30714	30714	30735
30735	30756	30756	30760	30760	30781	30781	30787	30787	30830	30830	30875
30875	30879	30879	30888	30888	30909	30909	30926	30926	30987	30987	30998
30998	31002	31002	31009	31009	31014	31014	31021	31021	31026	31026	31049
31049	31062	31062	31112	31112	31227	31227	31230	31230	31241	31241	31349
31349	31442	31442	31456	31456	31494	31495	31498	31498	31505	31505	31514
31514	31522	31554	31557	31588	31588	31604	31635	31650	31650	31688	31688
31704	31704	31708	31708	31719	31745	31745	31926	31926	32044	32044	32082
32082	32100	32100	32124	32124	32131	32131	32163	32163	32172	32172	32372
32372	32379	32379	32385	32385	32396	32396	32400	32400	32404	32404	32420
32420	32423	32423	32426	32426	32438	32438	32459	32459	32478	32478	32482
32482	32494	32494	32511	32511	32545	32545	32564	32564	32567	32567	32570
32570	32574	32574	32591	32591	32595	32595	32599	32599	32603	32603	32607
32607	32611	32611	32615	32615	32625	32625	32629	32629	32635	32635	32639
32639	32645	32645	32649	32649	32655	32655	32659	32659	32664	32664	32667
32667	32671	32671	32687	32687	32692	32702	32702	32707	32707	32747	32747
32791	32791	32815	32815	32823	32823	32829	32829	32833	32833	32837	32837
32862	32862	32880	32880	32884	32884	32903	32903	32916	32916	32920	32920
32938	32938	32952	32952	32984	32984	32992	32992	32996	32996	33019	33019
33027	33027	33037	33037	33051	33051	33061	33061	33065	33065	33128	33128
33150	33150	33467	33467	33484	33484	33527	33527	33535	33535	33538	33538
33542	33542	33545	33545	33559	33575	33575	33579	33579	33610	33611	33611
33615	33615	33615	33619	33619	33629	33629	33639	33639	33639	33649	33650
33650	33650	33654	33654	33660	33660	33669	33669	33673	33673	33677	33677
33820	33820	33831	33831	33904	33904	33917	33917	33956	33956	33999	33999
34005	34005	34029	34029	34033	34033	34072	34072	34076	34076	34089	34089
34093	34093	34133	34133	34147	34147	34150	34150	34154	34154	34157	34157
34168	34168	34172	34172	34178	34178	34198	34198	34241	34241	34255	34260
34260	34264	34264	34268	34268	34275	34275	34294	34294	34301	34301	34319
34319	34351	34351	34363	34363	34380	34380	34394	34394	34405	34405	34441
34441	34458	34458	34498	34498	34505	34505	34561	34561	34570	34570	34610
34610	34658	34658	34679	34679	34682	34682	34701	34701	34704	34707	34707
34714	34714	34714	34838	34838	34864	34864	34891	34891	34900	34900	34904

34904	34908	34908	34913	34913	35014	35014	35031	35031	35064	35064	35067
35067	35079	35079	35083	35083	35091	35091	35107	35107	35111	35111	35115
35115	35119	35119	35123	35123	35127	35127	35176	35176	35186	35186	35205
35205	35252	35252	35268	35268	35272	35272	35288	35288	35315	35315	35328
35328	35331	35331	35363	35363	35369	35369	35372	35372	35376	35376	35382
35382	35385	35385	35390	35390	35394	35394	35402	35402	35405	35405	35408
35408	35412	35412	35418	35418	35437	35437	35456	35456	35460	35460	35469
35469	35469	35478	35478	35482	35482	35482	35489	35489	35493	35493	35498
35498	35503	35503	35512	35512	35516	35516	35522	35522	35526	35526	35532
35532	35536	35536	35567	35567	35571	35571	35576	35576	35585	35585	35590
35590	35599	35599	35640	35651	35660	35670	35670	35698	35698	35703	35703
35741	35741	35779	35779	35779	35783	35783	35798	35798	35802	35802	35814
35814	35818	35818	35822	35822	35826	35826	35834	35834	35838	35838	35842
35842	35850	35850	35853	35853	35857	35857	35861	35861	35873	35873	35897
35897	35905	35905	35914	35914	35919	35919	35923	35923	35927	35927	35931
35931	35935	35935	35939	35939	35987	35987	36000	36000	36004	36004	36007
36007	36047	36047	36053	36053	36060	36060	36063	36063	36083	36083	36088
36088	36092	36092	36097	36097	36101	36101	36106	36106	36112	36112	36116
36116	36121	36121	36136	36136	36139	36139	36152	36152	36156	36156	36184
36184	36228	36228	36233	36233	36238	36238	36242	36242	36248	36248	36248
36251	36251	36289	36289	36304	36304	36316	36316	36337	36337	36345	36350
36354	36354	36479	36479	36496	36496	36563	36563	36580	36580	36615	36615
36619	36619	36619	36627	36627	36667	36667	36679	36679	36684	36684	36701
36701	36715	36715	36732	36732	36745	36745	36755	36755	36782	36782	36796
36799	36799	36809	36809	36841	36841	36849	36849	36853	36853	36871	36871
36882	36882	36896	36905	36905	36908	36908	36916	36916	37099	37099	37105
37105	37110	37110	37115	37115	37121	37121	37131	37131	37137	37137	37182
37189	37196	37196	37205	37205	37213	37219	37248	37248	37252	37252	37256
37256	37271	37271	37275	37275	37279	37279	37283	37283	37312	37312	37317
37317	37322	37322	37336	37375	37375	37380	37380	37384	37384	37388	37388
37400	37400	37428	37428	37432	37432	37437	37437	37441	37441	37488	37488
37537	37537	37560	37560	37563	37563	37586	37586	37629	37629	37635	37635
37641	37641	37647	37647	37663	37663	37672	37672	37686	37686	37694	37694
37698	37698	37719	37719	37723	37723	37727	37727	37737	37737	37744	37744
37770	37770	37789	37789	37801	37801	37807	37807	37813	37813	37831	37831
37839	37839	37845	37845	37853	37853	37857	37857	37868	37868	37878	37878
37882	37882	37914	37914	37958	37958	37962	37962	37965	37965	37969	37969
37973	37973	37977	37977	38011	38011	38065	38065	38084	38084	38092	38092
38095	38095	38106	38106	38109	38109	38113	38113	38152	38152	38157	38157
38174	38174	38202	38202	38206	38206	38214	38214	38217	38217	38222	38222
38225	38225	38229	38229	38272	38272	38279	38279	38283	38283	38346	38346
38351	38351	38370	38370	38379	38379	38385	38385	38390	38390	38405	38405
38419	38419	38428	38428	38434	38434	38444	38444	38473	38473	38529	38541
38547	38547	38552	38552	38556	38563	38563	38572	38572	38579	38579	38610
38610	38627	38627	38632	38632	38638	38638	38644	38644	38655	38655	38699
38699	38706	38706	38712	38712	38717	38717	38722	38722	38730	38730	38740
38740	38746	38746	38752	38752	38757	38757	38767	38767	38807	38807	38833
38834	38852	38852	38856	38856	38918	38918	38928	38928	39039	39039	39120
39120	39158	39158	39210	39210	39269	39269	39276	39276	39376	39376	39386
39386	39430	39430	39469	39469	39485	39485	39726	39726	39740	39740	39760
39760	39806	39806	39813	39813	39834	39834	39925	39925	39932	39932	39949
39949	39960	39960	40087	40087	40091	40091	40150	40150	40154	40154	40169
40169	40173	40173	40180	40180	40191	40191	40195	40195	40218	40218	40226
40226	40232	40232	40237	40237	40241	40241	40303	40303	40309	40309	40324
40324	40327	40327	40408	40408	40432	40432	40436	40436	40483	40484	40488

Cross Reference Listing - Field Names and Defined Values

	40489	40493	40494	40498	40498	40503	40503	40509	40509	40515	40515	40528
	40528	40542	40542	40548	40548	40552	40552	40561	40561	40572	40572	40577
	40577	40590	40590	40594	40594	40597	40597	40605	40605	40622	40622	40626
	40626	40647	40647	40651	40651	40858	41032	41032	41070	41070	41210	41210
	41228	41228	41310	41310	41337	41337	41351	41355	41355	41362	41367	41367
	41385	41413	41429	41429	41443	41561	41643	41643	41643	41648	41648	41651
	41651	41664	41664	41664	41669	41669	41673	41673	41679	41679	41683	41683
	41693	41693	41693	41698	41698	41698	41703	41703	41703	41953	41954	41958
	41959	41963	41964	41985	41986	42041	42045	42045	42053	42053	42057	42057
	42061	42061	42087	42087	42098	42099	42103	42103	42108	42109	42114	42114
	42119	42125	42129	42134	42138	42171	42176	42176	42346	42486	42486	42508
	42662	42662	42670	42674	42674	42683	42695	42695	42789	42789	42939	42939
	43080	43080	43192	43192	44751	44751	45169	45169	45604	45604	45674	45674
	45786	45786	46148	46148	46152	46152	46156	46156	46162	46162	46166	46166
	46179	46179	46183	46183	46201	46201	46219	46219	46243	46243	46251	46251
	46280	46280	46305	46305	46313	46313	46344	46344	46353	46353	46368	46368
	46368	46387	46387	46391	46391	46403	46403	46407	46407	46416	46416	46428
	46428	46432	46432	46441	46441	46453	46453	46457	46457	46475	46475	46478
	46478	46488	46488	46492	46492	46544	46544	46569	46569	46576	46576	46583
	46583	46590	46590	46595	46595	46601	46601	46608	46608	46615	46615	46622
	46622	46629	46629	46635	46635	46642	46642	46648	46648	46655	46655	46662
	46662	46668	46668	46675	46675	46682	46682	46689	46689	46696	46696	46716
	46716	46725	46725	46730	46731	46731	46740	46740	46744	46744	46748	46749
	46749	46763	46763	46781	46781	46807	46807	46810	46810	46815	46815	46819
	46819	46827	46827	46832	46832	46875	46875	46886	46886	46893	46893	46895
	46895	46908	46908	46911	46911	46922	46922	46926	46926	46947	46947	46952
	46952	46990	46990	46993	46993	47000	47000	47003	47003	47012	47012	47022
	47022	47062	47062	47070	47070	47091	47091	47094	47094	47098	47098	47101
	47101	47109	47109	47112	47112	47136	47136	47139	47139	47154	47154	47158
	47158	47162	47162	47165	47165	47169	47169	47173	47173	47176	47176	47181
	47181	47187	47187	47190	47190	47197	47197	47201	47201	47208	47208	47211
	47211	47215	47215	47219	47219	47222	47222	47227	47227	47230	47230	47237
	47237	47241	47241	47248	47248	47251	47251	47255	47255	47259	47259	47262
	47262	47265	47265	47271	47271	47281	47281	47295	47295	47299	47299	47302
	47302	47337	47338	47338	47338	47356	47356	47370	47371	47371	47371	47378
	47378	47393	47393	47402	47402	47406	47406	47412	47412	47415	47415	47423
	47423	47431	47431	47470	47470	47474	47474	47478	47478	47495	47495	47500
	47500	47505	47505	47510	47510	47515	47515	47520	47520	47541	47541	47546
	47546	47551	47551	47561	47561	47566	47570	47575	47575	47647	47647	47651
	47651	47655	47655	47661	47661	47740	47740	47751	47752	47759	47759	47782
	47782	47805	47805	47810	47810	47899	47899	47904	47904	47937	47937	47943
	47943	47947	47947	47984	47984	48033	48033	48084	48084	48087	48087	48090
	48090	48093	48093	48096	48096	48099	48099					
CCBRO.SRKSTA0	2813 #	5328	5328	5335	5335	5358	5358	5366	5366	5385	5385	5399
	5399	5403	5403	5409	5409	5455	5455	5459	5459	5463	5463	5472
	5472	5477	5477	5488	5488	5496	5496	5500	5500	5504	5504	5508
	5508	5515	5515	5519	5519	5523	5523	5531	5531	5535	5535	5540
	5540	5560	5560	5583	5583	5591	5591	5601	5601	5605	5605	5614
	5614	5617	5617	5626	5626	5630	5630	5659	5659	5662	5662	5676
	5676	5682	5682	5687	5687	5699	5699	5712	5712	5716	5716	5723
	5723	5733	5733	5750	5750	5785	5785	5796	5796	5824	5824	5899
	5899	5913	5913	5920	5920	5934	5934	5940	5940	5943	5943	5951
	5951	5954	5954	5957	5957	5963	5963	5968	5968	5978	5978	5992
	5992	5998	5998	6007	6007	6128	6128	6132	6132	6142	6142	6149
	6149	6156	6156	6160	6160	6171	6171	6174	6174	6221	6221	6225

6225	6229	6229	6233	6233	6236	6236	6240	6240	6244	6244	6248
6248	6256	6256	6265	6265	6270	6270	6274	6274	6281	6281	6285
6285	6289	6289	6293	6293	6299	6299	6303	6303	6307	6307	6313
6313	6317	6317	6321	6321	6327	6327	6331	6331	6335	6335	6341
6341	6345	6345	6349	6349	6355	6355	6359	6359	6363	6363	6369
6369	6373	6373	6377	6377	6383	6383	6387	6387	6391	6391	6397
6397	6401	6401	6405	6405	6411	6411	6415	6415	6419	6419	6425
6425	6433	6433	6464	6464	6471	6471	6478	6478	6483	6487	6487
6503	6503	6508	6508	6528	6528	6536	6536	6542	6542	6545	6545
6557	6557	6613	6613	6662	6662	6667	6667	6695	6709	6709	6753
6753	6800	6803	6803	6825	6825	6832	6832	6858	6858	6932	6932
6965	6965	6978	6978	7014	7014	7033	7037	7037	7042	7042	7048
7048	7051	7051	7055	7055	7059	7059	7062	7062	7066	7066	7071
7071	7074	7074	7078	7078	7083	7083	7086	7086	7090	7090	7095
7095	7098	7098	7102	7102	7107	7107	7110	7110	7113	7166	7166
7170	7170	7173	7173	7177	7177	7184	7184	7187	7187	7231	7231
7235	7235	7264	7264	7268	7268	7276	7276	7280	7280	7284	7284
7310	7310	7314	7314	7322	7322	7330	7330	7338	7338	7345	7345
7353	7353	7361	7361	7369	7369	7373	7373	7378	7378	7383	7383
7405	7405	7410	7410	7420	7420	7433	7433	7438	7438	7454	7454
7459	7459	7466	7466	7476	7476	7482	7482	7499	7499	7504	7504
7521	7521	7526	7526	7533	7533	7543	7543	7548	7548	7557	7557
7561	7561	7565	7565	7569	7569	7578	7578	7581	7581	7591	7591
7595	7595	7615	7615	7627	7627	7635	7635	7639	7644	7644	7648
7648	7652	7652	7655	7664	7664	7667	7676	7676	7686	7690	7690
7706	7706	7714	7714	7718	7718	7729	7729	7741	7741	7746	7746
7750	7750	7753	7753	7757	7757	7762	7762	7769	7769	7782	7786
7789	7789	7792	7809	7809	7812	7829	7829	7833	7833	7850	7850
7861	7861	7873	7873	7895	7900	7908	7908	7912	7912	7915	7915
7920	7944	7944	7949	7949	7958	7958	7963	7963	7975	7975	7986
7986	8009	8009	8014	8014	8026	8026	8032	8032	8058	8058	8063
8063	8068	8068	8072	8072	8076	8076	8080	8080	8094	8094	8143
8143	8187	8187	8191	8191	8199	8199	8206	8206	8910	8927	8927
8950	9027	9039	9315	9324	9396	9396	9401	9401	9416	9420	9420
9425	9430	9430	9441	9449	9449	9459	9459	9559	9559	9570	9570
9670	9670	9690	9690	9694	9770	9770	9778	9778	9829	9830	10817
10817	10825	10825	10868	10868	10879	10879	10926	10926	10930	10930	10934
10934	11029	11029	11034	11034	11149	11149	11154	11154	11329	11329	11337
11340	11340	11353	11383	11499	11712	11771	11771	11782	11782	11799	11799
11813	11813	11885	11885	12035	12035	12042	12042	12224	12238	12238	12333
12333	12343	12343	12372	12372	12385	12385	12400	12400	12513	12513	12519
12519	12550	12550	12556	12562	12562	12580	12580	12594	12594	12603	12621
12621	12633	12637	12637	12642	12642	12700	12700	12706	12706	12712	12712
12716	12716	12721	12721	12726	12726	12731	12731	12734	12734	12858	12858
12878	12878	13085	13085	13469	13480	13480	13588	13588	13625	13625	13670
13670	13684	13684	13702	13702	13878	13878	14088	14088	14113	14113	14118
14118	14426	14426	14442	14442	14446	14446	14501	14501	14505	14505	14666
14666	14690	14690	14702	14702	14713	14713	14735	14735	14837	14837	14843
14843	14973	14973	15008	15008	15016	15016	15073	15073	15079	15079	15084
15084	15088	15088	15095	15095	15098	15098	15123	15123	15166	15166	15330
15335	15386	15390	15390	15418	15418	15422	15422	15426	15426	15690	15690
15711	15711	15742	15742	15777	15777	15812	15812	15830	15830	15847	15847
15876	15876	15880	15880	15889	15905	15905	15997	15997	16060	16060	16064
16064	16069	16069	16114	16114	16120	16120	16137	16137	16141	16141	16200
16200	16230	16230	16290	16290	16435	16435	16483	16483	16525	16525	16563

Cross Reference Listing - Field Names and Defined Values

16563	16574	16574	16599	16599	16604	16604	16624	16624	16668	16668	16991
16991	16997	17118	17118	17123	17123	17160	17160	17169	17169	17173	17173
17177	17177	17238	17238	17246	17474	17474	17531	17547	17562	17562	17566
17566	17592	17592	17610	17610	17634	17634	18402	18460	18465	18470	18664
19042	19042	19053	19053	19065	19065	19077	19077	19154	19154	19167	19167
19237	19237	19242	19242	19304	19311	19316	19349	19354	19359	19400	19405
19470	19470	19479	19479	19484	19484	19583	19583	20124	20124	20129	20129
20135	20135	20140	20140	20156	20156	20156	20177	20177	20200	20200	20211
20211	20244	20244	20253	20253	20262	20262	20270	20270	20279	20279	20288
20288	20297	20297	20305	20305	20314	20314	20323	20323	20332	20332	20340
20340	20350	20350	20355	20355	20360	20360	20365	20365	20370	20370	20375
20375	20379	20379	20384	20384	20389	20389	20394	20394	20399	20399	20404
20404	20409	20409	20429	20429	20438	20438	20450	20450	20455	20455	20465
20465	20469	20469	20491	20491	20495	20495	20499	20499	20536	20536	20540
20540	20545	20545	20548	20548	20553	20553	20573	20573	20608	20608	20614
20614	20618	20618	20623	20623	20627	20627	20633	20633	20637	20637	20643
20643	20647	20647	20653	20653	20657	20657	20663	20663	20667	20667	20672
20672	20676	20676	20682	20682	20686	20686	20692	20692	20696	20696	20702
20702	20706	20706	20712	20712	20716	20716	20721	20721	20742	20742	20751
20751	20780	20780	20788	20788	20792	20792	20806	20806	20811	20811	20901
20901	20914	20914	20930	20930	20950	20955	20955	20962	20962	20966	20966
20972	20972	20976	20976	21032	21046	21051	21102	21102	21117	21117	21137
21137	21144	21166	21166	21206	21206	21213	21213	21263	21263	21320	21320
21324	21324	21327	21327	21350	21386	21386	21396	21396	21412	21439	21439
21454	21454	21468	21468	21472	21472	21543	21543	21546	21546	21604	21604
21615	21615	21621	21621	21655	21655	21665	21665	21698	21698	21730	21730
21758	21758	21787	21787	21794	21794	21808	21808	21848	21848	21858	21858
21868	21868	21874	21874	21889	21889	21903	21903	21912	21912	21952	21952
21969	21969	21979	21979	21999	21999	22004	22004	22012	22012	22021	22021
22429	22429	22482	22482	22486	22486	22547	22547	22638	22638	22677	22677
22785	22785	23001	23001	23004	23004	23031	23031	23035	23035	23039	23039
23050	23050	23079	23079	23096	23096	23107	23107	23129	23129	23168	23168
23172	23172	23176	23176	23187	23187	23205	23205	23237	23237	23245	23245
23279	23279	23333	23333	23437	23437	23469	23479	23688	23688	23694	23761
23761	23808	23808	23818	23818	23922	23922	23947	23947	23957	23957	24015
24015	24062	24062	24103	24103	24251	24251	24314	24343	24367	24367	24423
24423	24463	24463	24472	24472	24477	24477	24486	24495	24495	24526	24526
24541	24541	24647	24647	24665	24665	24669	24669	24680	24680	24684	24684
24708	24708	24760	24760	24782	24782	24797	24797	24802	24802	24805	24805
24835	24835	24869	24869	24930	24930	24933	24933	24952	24952	24959	24959
25025	25025	25041	25041	25047	25047	25051	25051	25092	25092	25110	25110
25119	25119	25165	25165	25204	25204	25243	25243	25284	25284	25292	25292
25299	25299	25368	25368	25612	25612	25634	25634	25650	25650	25677	25677
25681	25681	25714	25714	25792	25792	25797	25797	25973	25973	25994	25994
26003	26003	26006	26006	26009	26009	26017	26017	26029	26029	26111	26111
26143	26143	26146	26146	26172	26212	26212	26293	26293	26361	26361	26364
26364	26370	26370	26373	26373	26465	26465	26501	26501	26504	26504	26583
26583	26790	26790	26794	26794	26848	26848	26853	26853	26896	26896	27014
27014	27017	27017	27023	27023	27026	27026	27044	27044	27051	27051	27063
27063	27067	27067	27102	27102	27333	27333	27375	27375	27379	27379	27499
27513	27526	27534	27665	27665	27687	27687	27732	27732	27775	27775	27779
27779	27797	27797	27801	27801	27817	27817	27827	27827	27835	27835	27839
27839	27845	27845	27849	27849	27855	27855	27859	27859	27868	27868	27872
27872	27876	27876	27911	27911	27916	27916	27984	27984	28008	28008	28027
28027	28049	28049	28065	28065	28075	28075	28106	28106	28117	28117	28144

28144	28147	28147	28155	28155	28178	28178	28183	28183	28188	28188	28213
28213	28245	28245	28249	28264	28264	28268	28268	28279	28285	28285	28298
28298	28312	28312	28324	28328	28328	28334	28334	28338	28338	28341	28341
28418	28418	28428	28428	28453	28453	28477	28477	28494	28494	28500	28500
28506	28506	28534	28534	28538	28538	28559	28563	28563	28574	28584	28589
28598	28598	28601	28601	28619	28619	28661	28661	28665	28665	28670	28670
28725	28746	28770	28770	28774	28774	28779	28779	28783	28783	28793	28793
28797	28797	28824	28824	28828	28828	28836	28836	28855	28855	29074	29074
29089	29089	29112	29112	29116	29116	29134	29134	29138	29138	29142	29142
29176	29176	29246	29246	29250	29250	29269	29269	29381	29381	29488	29488
29491	29491	29511	29511	29535	29535	29538	29538	29560	29560	29567	29567
29585	29585	29590	29590	29595	29595	29600	29600	29605	29605	29636	29636
29660	29660	29670	29670	29680	29680	29684	29684	29689	29698	29698	29712
29712	29717	29717	29723	29723	29727	29727	29731	29818	29818	29822	29822
29828	29828	29836	29836	29843	29843	29850	29850	29869	29869	29877	29877
29912	29912	29986	30029	30105	30105	30202	30202	30226	30226	30244	30244
30249	30249	30253	30253	30257	30257	30267	30267	30273	30273	30277	30277
30281	30281	30286	30286	30296	30296	30352	30352	30374	30374	30380	30380
30388	30388	30451	30451	30457	30457	30461	30461	30481	30481	30492	30492
30499	30499	30518	30518	30532	30532	30575	30581	30586	30586	30601	30601
30611	30611	30620	30620	30626	30626	30632	30632	30636	30636	30641	30641
30714	30714	30735	30735	30747	30756	30756	30760	30760	30781	30781	30787
30787	30826	30830	30875	30875	30879	30879	30888	30888	30909	30909	30915
30919	30926	30926	30987	30987	30998	30998	31002	31002	31009	31009	31014
31014	31021	31021	31026	31026	31049	31049	31062	31062	31112	31112	31227
31227	31230	31230	31241	31241	31349	31349	31442	31442	31456	31456	31495
31498	31498	31505	31505	31514	31514	31588	31588	31650	31650	31688	31688
31704	31704	31708	31708	31745	31745	31926	31926	32044	32044	32082	32082
32100	32100	32124	32124	32131	32131	32163	32163	32172	32172	32200	32203
32206	32211	32237	32240	32243	32248	32274	32277	32280	32285	32372	32372
32379	32379	32385	32385	32396	32396	32400	32400	32404	32404	32420	32420
32423	32423	32426	32426	32438	32438	32459	32459	32478	32478	32482	32482
32494	32494	32511	32511	32545	32545	32564	32564	32567	32567	32570	32570
32574	32574	32591	32591	32595	32595	32599	32599	32603	32603	32607	32607
32611	32611	32615	32615	32625	32625	32629	32629	32635	32635	32639	32639
32645	32645	32649	32649	32655	32655	32659	32659	32664	32664	32667	32667
32671	32671	32687	32687	32692	32702	32702	32707	32707	32747	32747	32791
32791	32815	32815	32823	32823	32829	32829	32833	32833	32837	32837	32862
32862	32880	32880	32884	32884	32903	32903	32916	32916	32920	32920	32938
32938	32952	32952	32984	32984	32992	32992	32996	32996	33019	33019	33027
33027	33037	33037	33051	33051	33061	33061	33065	33065	33128	33128	33150
33150	33208	33467	33467	33484	33484	33527	33527	33535	33535	33538	33538
33542	33542	33545	33545	33559	33563	33575	33575	33579	33579	33611	33611
33615	33615	33619	33619	33629	33629	33639	33639	33650	33650	33650	33654
33654	33660	33660	33669	33669	33673	33673	33677	33677	33820	33820	33831
33831	33904	33904	33917	33917	33956	33956	33999	33999	34005	34005	34029
34029	34033	34033	34072	34072	34076	34076	34089	34089	34093	34093	34133
34133	34147	34147	34150	34150	34154	34154	34157	34157	34168	34168	34172
34172	34178	34178	34198	34198	34241	34241	34260	34260	34264	34264	34268
34268	34275	34275	34294	34294	34301	34301	34319	34319	34351	34363	34363
34380	34380	34394	34394	34405	34405	34441	34441	34458	34458	34498	34498
34505	34505	34561	34561	34570	34570	34610	34610	34658	34658	34679	34679
34682	34682	34701	34701	34704	34707	34707	34714	34714	34838	34838	34864
34864	34891	34891	34900	34900	34904	34904	34908	34908	34913	34913	35014
35014	35031	35031	35064	35064	35067	35067	35079	35079	35083	35083	35091

Cross Reference Listing - Field Names and Defined Values

35091	35107	35107	35111	35111	35115	35115	35119	35119	35123	35123	35127
35127	35176	35176	35186	35186	35205	35205	35252	35252	35268	35268	35272
35288	35288	35315	35315	35328	35328	35331	35331	35363	35369	35369	35372
35372	35376	35376	35382	35382	35385	35385	35390	35390	35394	35394	35402
35402	35405	35405	35408	35408	35412	35412	35418	35418	35433	35437	35437
35456	35456	35460	35460	35469	35469	35478	35478	35482	35482	35489	35489
35493	35493	35498	35498	35503	35503	35512	35512	35516	35516	35522	35522
35526	35526	35532	35532	35536	35536	35567	35567	35571	35571	35576	35576
35585	35585	35590	35590	35599	35599	35640	35651	35660	35670	35670	35698
35698	35703	35703	35741	35741	35779	35779	35779	35783	35783	35798	35798
35802	35802	35814	35814	35818	35818	35822	35822	35826	35826	35834	35834
35838	35838	35842	35842	35850	35850	35853	35853	35857	35857	35861	35861
35873	35873	35897	35897	35905	35905	35914	35914	35919	35919	35923	35923
35927	35927	35931	35931	35935	35935	35939	35939	35954	35958	35987	35987
36000	36000	36004	36004	36007	36007	36043	36047	36047	36053	36053	36060
36060	36063	36063	36069	36083	36083	36088	36088	36092	36092	36097	36097
36101	36101	36106	36106	36112	36112	36116	36116	36121	36121	36128	36136
36136	36139	36139	36152	36152	36156	36156	36184	36184	36228	36228	36233
36233	36238	36238	36242	36242	36248	36248	36248	36251	36251	36289	36289
36304	36304	36316	36316	36337	36337	36345	36350	36354	36354	36479	36479
36496	36496	36563	36563	36580	36580	36615	36615	36619	36619	36619	36627
36627	36667	36667	36679	36679	36684	36684	36701	36701	36715	36715	36732
36732	36745	36745	36755	36755	36782	36782	36796	36799	36799	36809	36809
36841	36841	36849	36849	36853	36853	36871	36871	36882	36882	36896	36905
36905	36908	36908	36916	36916	37099	37099	37105	37105	37110	37110	37115
37115	37121	37121	37131	37131	37137	37137	37182	37189	37196	37196	37205
37205	37213	37219	37248	37248	37252	37252	37256	37256	37271	37271	37275
37275	37279	37279	37283	37283	37312	37312	37317	37317	37322	37322	37336
37375	37375	37380	37380	37384	37384	37388	37388	37400	37400	37428	37428
37432	37432	37437	37437	37441	37441	37488	37488	37537	37537	37560	37560
37563	37563	37586	37586	37629	37629	37635	37635	37641	37641	37647	37647
37663	37663	37672	37672	37686	37686	37694	37694	37698	37698	37719	37719
37723	37723	37727	37727	37737	37737	37744	37744	37770	37770	37789	37789
37801	37801	37807	37807	37813	37813	37831	37831	37839	37839	37845	37845
37853	37853	37857	37857	37868	37868	37878	37878	37882	37882	37914	37914
37958	37958	37962	37962	37965	37965	37969	37969	37973	37973	37977	37977
38011	38011	38065	38065	38084	38084	38092	38092	38095	38095	38106	38106
38109	38109	38113	38113	38152	38152	38157	38157	38174	38174	38202	38202
38206	38206	38214	38214	38217	38217	38222	38222	38225	38225	38229	38229
38272	38272	38279	38279	38283	38283	38346	38346	38351	38351	38370	38370
38379	38379	38385	38385	38390	38390	38405	38405	38419	38419	38428	38428
38434	38434	38444	38444	38473	38473	38529	38541	38547	38547	38552	38552
38556	38563	38563	38572	38572	38579	38579	38610	38610	38627	38627	38632
38632	38638	38638	38644	38644	38655	38655	38699	38699	38706	38706	38712
38712	38717	38717	38722	38722	38730	38730	38740	38740	38746	38746	38752
38752	38757	38757	38767	38767	38807	38807	38834	38852	38852	38856	38856
38918	38918	38928	38928	38953	38971	38975	39039	39039	39120	39120	39158
39158	39210	39210	39269	39269	39276	39276	39376	39376	39386	39386	39430
39430	39469	39469	39485	39485	39726	39726	39740	39740	39760	39760	39806
39806	39813	39813	39834	39834	39925	39925	39932	39932	39949	39949	39960
39960	40087	40087	40091	40091	40150	40150	40154	40154	40169	40169	40173
40173	40180	40180	40191	40191	40195	40195	40218	40218	40226	40226	40232
40232	40237	40237	40241	40241	40300	40303	40303	40309	40309	40324	40324
40327	40327	40408	40408	40432	40432	40436	40436	40484	40484	40489	40489
40494	40494	40498	40498	40503	40503	40506	40509	40509	40515	40515	40528

Cross Reference Listing - Field Names and Defined Values

	40528	40532	40538	40542	40542	40548	40548	40552	40552	40561	40561	40572
	40572	40577	40577	40590	40590	40594	40594	40597	40597	40605	40605	40622
	40622	40626	40626	40630	40647	40647	40651	40651	40858	41032	41032	41070
	41070	41210	41210	41228	41228	41310	41310	41337	41337	41351	41355	41355
	41362	41367	41367	41385	41429	41429	41443	41561	41643	41643	41643	41648
	41648	41651	41651	41664	41664	41664	41669	41669	41673	41673	41679	41679
	41683	41683	41693	41693	41693	41698	41698	41698	41703	41703	41703	41954
	41959	41964	41986	42041	42045	42045	42053	42053	42057	42057	42061	42061
	42087	42087	42099	42103	42109	42114	42114	42129	42138	42171	42176	42176
	42486	42486	42508	42662	42662	42670	42674	42674	42683	42695	42695	42789
	42789	42939	42939	43080	43080	43192	43192	44751	44751	45058	45169	45169
	45604	45604	45674	45674	45786	45786	46148	46148	46152	46152	46156	46156
	46162	46162	46166	46166	46179	46179	46183	46183	46201	46201	46219	46219
	46243	46243	46251	46251	46280	46280	46305	46305	46313	46313	46344	46344
	46353	46353	46368	46368	46387	46387	46391	46391	46403	46403	46407	46407
	46416	46416	46428	46428	46432	46432	46441	46441	46453	46453	46457	46457
	46475	46475	46478	46478	46488	46488	46492	46492	46544	46544	46569	46569
	46576	46576	46583	46583	46590	46590	46595	46595	46601	46601	46608	46608
	46615	46615	46622	46622	46629	46629	46635	46635	46642	46642	46648	46648
	46655	46655	46662	46662	46668	46668	46675	46675	46682	46682	46689	46689
	46696	46696	46716	46716	46725	46725	46731	46731	46740	46740	46744	46744
	46749	46749	46763	46763	46781	46781	46807	46807	46810	46810	46815	46815
	46819	46819	46827	46827	46832	46832	46875	46875	46886	46886	46893	46893
	46895	46895	46908	46911	46911	46922	46922	46926	46926	46947	46947	46952
	46952	46990	46993	46993	47000	47003	47003	47012	47012	47022	47022	47062
	47062	47070	47070	47091	47091	47094	47094	47098	47098	47101	47101	47109
	47109	47112	47112	47136	47136	47139	47139	47154	47154	47158	47158	47162
	47162	47165	47165	47169	47169	47173	47173	47176	47176	47181	47181	47187
	47187	47190	47190	47197	47197	47201	47201	47208	47208	47211	47211	47215
	47215	47219	47219	47222	47222	47227	47227	47230	47230	47237	47237	47241
	47241	47248	47248	47251	47251	47255	47255	47259	47259	47262	47262	47265
	47265	47271	47271	47281	47281	47295	47295	47299	47299	47302	47302	47338
	47338	47338	47356	47356	47371	47371	47371	47378	47378	47393	47393	47402
	47402	47406	47406	47412	47412	47415	47415	47423	47423	47431	47431	47470
	47470	47474	47474	47478	47478	47495	47495	47500	47500	47505	47505	47510
	47510	47515	47515	47520	47520	47541	47541	47546	47546	47551	47551	47561
	47561	47566	47570	47575	47575	47647	47647	47651	47651	47655	47655	47661
	47661	47740	47740	47752	47759	47759	47782	47782	47805	47805	47810	47810
	47899	47899	47904	47904	47937	47937	47943	47943	47947	47947	47984	47984
	48033	48033	48084	48084	48087	48087	48090	48090	48093	48093	48096	48096
	48099	48099										
CCBR1.CCBR0.IR0	2812 #	5328	5328	5335	5335	5358	5358	5366	5366	5385	5385	5399
	5399	5403	5403	5409	5409	5455	5455	5459	5459	5463	5463	5472
	5472	5477	5477	5488	5488	5496	5496	5500	5500	5504	5504	5508
	5508	5515	5515	5519	5519	5523	5523	5531	5531	5535	5535	5540
	5540	5560	5560	5583	5583	5591	5591	5601	5601	5605	5605	5614
	5614	5617	5617	5626	5626	5630	5630	5659	5659	5662	5662	5676
	5676	5682	5682	5687	5687	5699	5699	5712	5712	5716	5716	5723
	5723	5733	5733	5750	5750	5785	5785	5796	5796	5824	5824	5899
	5899	5913	5913	5920	5920	5934	5934	5940	5940	5943	5943	5951
	5951	5954	5954	5957	5957	5963	5963	5968	5968	5978	5978	5992
	5992	5998	5998	6007	6007	6128	6128	6132	6132	6142	6142	6149
	6149	6156	6156	6160	6160	6171	6171	6174	6174	6221	6221	6225
	6225	6229	6229	6233	6233	6236	6236	6240	6240	6244	6244	6248
	6248	6256	6256	6265	6265	6270	6270	6274	6274	6281	6281	6285

Cross Reference Listing - Field Names and Defined Values

6285	6289	6289	6293	6293	6299	6299	6303	6303	6307	6307	6313
6313	6317	6317	6321	6321	6327	6327	6331	6331	6335	6335	6341
6341	6345	6345	6349	6349	6355	6355	6359	6359	6363	6363	6369
6369	6373	6373	6377	6377	6383	6383	6387	6387	6391	6391	6397
6397	6401	6401	6405	6405	6411	6411	6415	6415	6419	6419	6425
6425	6433	6433	6464	6464	6471	6471	6478	6478	6487	6487	6503
6503	6508	6508	6528	6528	6536	6536	6542	6542	6545	6545	6557
6557	6613	6613	6662	6662	6667	6667	6695	6709	6709	6753	6753
6803	6803	6825	6825	6832	6832	6858	6858	6932	6932	6965	6965
6978	6978	7014	7014	7033	7037	7037	7042	7042	7048	7048	7051
7051	7055	7055	7059	7059	7062	7062	7066	7066	7071	7071	7074
7074	7078	7078	7083	7083	7086	7086	7090	7090	7095	7095	7098
7098	7102	7102	7107	7107	7110	7110	7166	7166	7170	7170	7173
7173	7177	7177	7184	7184	7187	7187	7231	7231	7235	7235	7264
7264	7268	7268	7276	7276	7280	7280	7284	7284	7310	7310	7314
7314	7322	7322	7330	7330	7338	7338	7345	7345	7353	7353	7361
7361	7369	7369	7373	7373	7378	7378	7383	7383	7405	7405	7410
7410	7420	7420	7433	7433	7438	7438	7454	7454	7459	7459	7466
7466	7476	7476	7482	7482	7499	7499	7504	7504	7521	7521	7526
7526	7533	7533	7543	7543	7548	7548	7557	7557	7561	7561	7565
7565	7569	7569	7578	7578	7581	7581	7591	7591	7595	7595	7615
7615	7627	7627	7635	7635	7644	7644	7648	7648	7652	7652	7664
7664	7676	7676	7690	7690	7706	7706	7714	7714	7718	7718	7729
7729	7741	7741	7746	7746	7750	7750	7753	7753	7757	7757	7762
7762	7769	7769	7789	7789	7809	7809	7829	7829	7833	7833	7850
7850	7861	7861	7873	7873	7908	7908	7912	7912	7915	7915	7944
7944	7949	7949	7958	7958	7963	7963	7975	7975	7986	7986	8009
8009	8014	8014	8026	8026	8032	8032	8058	8058	8063	8063	8068
8068	8072	8072	8076	8076	8080	8080	8094	8094	8143	8143	8187
8187	8191	8191	8199	8199	8206	8206	8910	8927	8950	9027	9039
9315	9324	9396	9396	9401	9401	9416	9420	9420	9425	9430	9430
9441	9449	9449	9459	9459	9559	9559	9570	9570	9670	9670	9690
9690	9694	9770	9770	9778	9778	9830	10817	10817	10825	10825	10868
10868	10879	10879	10926	10926	10930	10930	10934	10934	11029	11029	11034
11034	11149	11149	11154	11154	11329	11329	11337	11340	11340	11353	11499
11712	11771	11771	11782	11782	11799	11799	11813	11813	11885	11885	12035
12035	12042	12042	12224	12238	12238	12333	12333	12343	12343	12372	12372
12385	12385	12400	12400	12513	12513	12519	12519	12550	12550	12556	12562
12562	12580	12580	12594	12594	12621	12621	12633	12637	12637	12642	12642
12700	12700	12706	12706	12712	12712	12716	12716	12721	12721	12726	12726
12731	12731	12734	12734	12858	12858	12878	12878	13085	13085	13469	13480
13480	13588	13588	13625	13625	13670	13670	13684	13684	13702	13702	13878
13878	14088	14088	14113	14113	14118	14118	14426	14426	14442	14442	14446
14446	14501	14501	14505	14505	14666	14666	14690	14690	14702	14702	14713
14713	14735	14735	14837	14837	14843	14843	14973	14973	15008	15008	15016
15016	15073	15073	15079	15079	15084	15084	15088	15088	15095	15095	15098
15098	15123	15123	15166	15166	15330	15335	15386	15390	15390	15418	15418
15422	15422	15426	15426	15690	15690	15711	15711	15742	15742	15777	15777
15812	15812	15830	15830	15847	15847	15876	15876	15880	15880	15889	15905
15905	15997	15997	16060	16060	16064	16064	16069	16069	16114	16114	16120
16120	16137	16137	16141	16141	16200	16200	16230	16230	16290	16290	16435
16435	16483	16483	16525	16525	16563	16563	16574	16574	16599	16599	16604
16604	16624	16624	16668	16668	16991	16991	16997	17118	17118	17123	17123
17160	17160	17169	17169	17173	17173	17177	17177	17238	17238	17246	17474
17474	17562	17562	17566	17566	17592	17592	17610	17610	17634	17634	19042

Cross Reference Listing - Field Names and Defined Values

19042	19053	19053	19065	19065	19077	19077	19154	19154	19167	19167	19237
19237	19242	19242	19304	19311	19316	19349	19354	19359	19399	19400	19405
19405	19450	19456	19470	19470	19479	19479	19484	19484	19583	19533	20124
20124	20129	20129	20135	20135	20140	20140	20156	20156	20156	20177	20177
20200	20200	20211	20211	20244	20244	20253	20253	20262	20262	20270	20270
20279	20279	20288	20288	20297	20297	20305	20305	20314	20314	20323	20323
20332	20332	20340	20340	20350	20350	20355	20355	20360	20360	20365	20365
20370	20370	20375	20375	20379	20379	20384	20384	20389	20389	20394	20394
20399	20399	20404	20404	20409	20409	20429	20429	20438	20438	20450	20450
20455	20455	20465	20465	20469	20469	20491	20491	20495	20495	20499	20499
20536	20536	20540	20540	20545	20545	20548	20548	20553	20553	20573	20573
20608	20608	20614	20614	20618	20618	20623	20623	20627	20627	20633	20633
20637	20637	20643	20643	20647	20647	20653	20653	20657	20657	20663	20663
20667	20667	20672	20672	20676	20676	20682	20682	20686	20686	20692	20692
20696	20696	20702	20702	20706	20706	20712	20712	20716	20716	20721	20721
20742	20742	20751	20751	20780	20780	20788	20788	20792	20792	20806	20806
20811	20811	20901	20901	20914	20914	20930	20930	20950	20955	20955	20962
20962	20966	20966	20972	20972	20976	20976	21032	21046	21051	21102	21102
21117	21117	21137	21137	21144	21166	21166	21206	21206	21213	21213	21263
21263	21320	21320	21324	21324	21327	21327	21350	21386	21386	21396	21396
21412	21439	21439	21454	21454	21468	21468	21472	21472	21543	21543	21546
21546	21604	21604	21615	21615	21621	21621	21655	21655	21665	21665	21698
21698	21730	21730	21758	21758	21787	21787	21794	21794	21808	21808	21848
21848	21858	21858	21868	21868	21874	21874	21889	21889	21903	21903	21912
21912	21952	21952	21969	21969	21979	21979	21999	21999	22004	22004	22012
22012	22021	22021	22429	22429	22482	22482	22486	22486	22547	22547	22638
22638	22677	22677	22785	22785	23001	23001	23004	23004	23031	23031	23035
23035	23039	23039	23050	23050	23079	23079	23096	23096	23107	23107	23129
23129	23168	23168	23172	23172	23176	23176	23187	23187	23205	23205	23237
23237	23245	23245	23279	23279	23333	23333	23437	23437	23469	23479	23688
23688	23694	23761	23761	23808	23808	23818	23818	23922	23922	23947	23947
23957	23957	24015	24015	24062	24062	24103	24103	24251	24251	24314	24343
24367	24367	24423	24423	24463	24463	24472	24472	24477	24477	24486	24495
24495	24526	24526	24541	24541	24647	24647	24665	24665	24669	24669	24680
24680	24684	24684	24708	24708	24760	24760	24782	24782	24797	24797	24802
24802	24805	24805	24835	24835	24869	24869	24930	24930	24933	24933	24952
24952	24959	24959	25025	25025	25041	25041	25047	25047	25051	25051	25092
25092	25110	25110	25119	25119	25165	25165	25204	25204	25243	25243	25284
25284	25292	25292	25299	25299	25368	25368	25612	25612	25634	25634	25650
25650	25677	25677	25681	25681	25714	25714	25792	25792	25797	25797	25973
25973	25994	25994	26003	26003	26006	26006	26009	26009	26017	26017	26029
26029	26111	26111	26143	26143	26146	26146	26172	26212	26212	26293	26293
26361	26361	26364	26364	26370	26370	26373	26373	26465	26465	26501	26501
26504	26504	26583	26583	26790	26790	26794	26794	26848	26848	26853	26853
26896	26896	27014	27014	27017	27017	27023	27023	27026	27026	27044	27044
27051	27051	27063	27063	27067	27067	27102	27102	27353	27353	27375	27375
27379	27379	27665	27665	27687	27687	27732	27732	27775	27775	27779	27779
27797	27797	27801	27801	27817	27817	27827	27827	27835	27835	27839	27839
27845	27845	27849	27849	27855	27855	27859	27859	27868	27868	27872	27872
27876	27876	27911	27911	27916	27916	27984	27984	28008	28008	28027	28027
28049	28049	28065	28065	28075	28075	28106	28106	28117	28117	28144	28144
28147	28147	28155	28155	28178	28178	28183	28183	28186	28188	28213	28213
28245	28245	28249	28264	28264	28268	28268	28279	28285	28285	28298	28298
28312	28312	28324	28328	28328	28334	28334	28338	28338	28341	28341	28418
28418	28428	28428	28453	28453	28477	28477	28494	28494	28500	28500	28506

Cross Reference Listing - Field Names and Defined Values

28506	28534	28534	28538	28538	28563	28563	28584	28589	28598	28598	28601
28601	28619	28619	28661	28661	28665	28665	28670	28670	28770	28770	28774
28774	28779	28779	28783	28783	28793	28793	28797	28797	28824	28824	28828
28828	28836	28836	28855	28855	29074	29074	29089	29089	29112	29112	29116
29116	29134	29134	29138	29138	29142	29142	29176	29176	29246	29246	29250
29250	29269	29269	29381	29381	29488	29488	29491	29491	29511	29511	29535
29535	29538	29538	29560	29560	29567	29567	29585	29585	29590	29590	29595
29595	29600	29600	29605	29605	29636	29636	29660	29660	29670	29670	29680
29680	29684	29684	29689	29698	29698	29712	29712	29717	29717	29723	29723
29727	29727	29731	29818	29818	29822	29822	29828	29828	29836	29836	29843
29843	29850	29850	29869	29869	29877	29877	29912	29912	29986	30029	30105
30105	30202	30202	30226	30226	30244	30244	30249	30249	30253	30253	30257
30257	30267	30267	30273	30273	30277	30277	30281	30281	30286	30286	30296
30296	30352	30352	30374	30374	30380	30380	30388	30388	30451	30451	30457
30457	30461	30461	30481	30481	30492	30492	30499	30499	30518	30518	30532
30532	30575	30581	30586	30586	30601	30601	30611	30611	30620	30620	30626
30626	30632	30632	30636	30636	30641	30641	30714	30714	30735	30735	30756
30756	30760	30760	30781	30781	30787	30787	30830	30875	30875	30879	30879
30888	30888	30909	30909	30926	30926	30987	30987	30998	30998	31002	31002
31009	31009	31014	31014	31021	31021	31026	31026	31049	31049	31062	31062
31112	31112	31227	31227	31230	31230	31241	31241	31349	31349	31442	31442
31456	31456	31495	31498	31498	31505	31505	31514	31514	31588	31588	31650
31650	31688	31688	31704	31704	31708	31708	31745	31745	31926	31926	32044
32044	32082	32082	32100	32100	32124	32124	32131	32131	32163	32163	32172
32172	32372	32372	32379	32379	32385	32385	32396	32396	32400	32400	32404
32404	32420	32420	32423	32423	32426	32426	32438	32438	32459	32459	32478
32478	32482	32482	32494	32494	32511	32511	32545	32545	32564	32564	32567
32567	32570	32570	32574	32574	32591	32591	32595	32595	32599	32599	32603
32603	32607	32607	32611	32611	32615	32615	32625	32625	32629	32629	32635
32635	32639	32639	32645	32645	32649	32649	32655	32655	32659	32659	32664
32664	32667	32667	32671	32671	32687	32687	32692	32702	32702	32707	32707
32747	32747	32791	32791	32815	32815	32823	32823	32829	32829	32833	32833
32837	32837	32862	32862	32880	32880	32884	32884	32903	32903	32916	32916
32920	32920	32938	32938	32952	32952	32984	32984	32992	32992	32996	32996
33019	33019	33027	33027	33037	33037	33051	33051	33061	33061	33065	33065
33128	33128	33150	33150	33467	33467	33484	33484	33527	33527	33535	33535
33538	33538	33542	33542	33545	33545	33559	33575	33575	33579	33579	33611
33611	33615	33615	33619	33619	33629	33629	33639	33639	33650	33650	33650
33654	33654	33660	33660	33669	33669	33673	33673	33677	33677	33820	33820
33831	33831	33904	33904	33917	33917	33956	33956	33999	33999	34005	34005
34029	34029	34033	34033	34072	34072	34076	34076	34089	34089	34093	34093
34133	34133	34147	34147	34150	34150	34154	34154	34157	34157	34168	34168
34172	34172	34178	34178	34198	34198	34241	34241	34260	34260	34264	34264
34268	34268	34275	34275	34294	34294	34301	34301	34319	34319	34351	34363
34363	34380	34380	34394	34394	34405	34405	34441	34441	34458	34458	34498
34498	34505	34505	34561	34561	34570	34570	34610	34610	34658	34658	34679
34679	34682	34682	34701	34701	34704	34707	34707	34714	34714	34838	34838
34864	34864	34891	34891	34900	34900	34904	34904	34908	34908	34913	34913
35014	35014	35031	35031	35064	35064	35067	35067	35079	35079	35083	35083
35091	35091	35107	35107	35111	35111	35115	35115	35119	35119	35123	35123
35127	35127	35176	35176	35186	35186	35205	35205	35252	35252	35268	35268
35272	35288	35288	35315	35315	35328	35328	35331	35331	35363	35369	35369
35372	35372	35376	35376	35382	35382	35385	35385	35390	35390	35394	35394
35402	35402	35405	35405	35408	35408	35412	35412	35418	35418	35437	35437
35456	35456	35460	35460	35469	35469	35478	35478	35482	35482	35489	35489

35493	35493	35498	35498	35503	35503	35512	35512	35516	35516	35522	35522
35526	35526	35532	35532	35536	35536	35567	35567	35571	35571	35576	35576
35585	35585	35590	35590	35599	35599	35640	35651	35660	35670	35670	35698
35698	35703	35703	35741	35741	35779	35779	35779	35783	35783	35798	35798
35802	35802	35814	35814	35818	35818	35822	35822	35826	35826	35834	35834
35838	35838	35842	35842	35850	35850	35853	35853	35857	35857	35861	35861
35873	35873	35897	35897	35905	35905	35914	35914	35919	35919	35923	35923
35927	35927	35931	35931	35935	35935	35939	35939	35987	35987	36000	36000
36004	36004	36007	36007	36047	36047	36053	36053	36060	36060	36063	36063
36083	36083	36088	36088	36092	36092	36097	36097	36101	36101	36106	36106
36112	36112	36116	36116	36121	36121	36136	36136	36139	36139	36152	36152
36156	36156	36184	36184	36228	36228	36233	36233	36238	36238	36242	36242
36248	36248	36248	36251	36251	36289	36289	36304	36304	36316	36316	36337
36337	36345	36350	36354	36354	36479	36479	36496	36496	36563	36563	36580
36580	36615	36615	36619	36619	36619	36627	36627	36667	36667	36679	36679
36684	36684	36701	36701	36715	36715	36732	36732	36745	36745	36755	36755
36782	36782	36796	36799	36799	36809	36809	36841	36841	36849	36849	36853
36853	36871	36871	36882	36882	36896	36905	36905	36908	36908	36916	36916
37099	37099	37105	37105	37110	37110	37115	37115	37121	37121	37131	37131
37137	37137	37182	37189	37196	37196	37205	37205	37213	37219	37248	37248
37252	37252	37256	37256	37271	37271	37275	37275	37279	37279	37283	37283
37312	37312	37317	37317	37322	37322	37336	37375	37375	37380	37380	37384
37384	37388	37388	37400	37400	37428	37428	37432	37432	37437	37437	37441
37441	37488	37488	37537	37537	37560	37560	37563	37563	37586	37586	37629
37629	37635	37635	37641	37641	37647	37647	37663	37663	37672	37672	37686
37686	37694	37694	37698	37698	37719	37719	37723	37723	37727	37727	37737
37737	37744	37744	37770	37770	37789	37789	37801	37801	37807	37807	37813
37813	37831	37831	37839	37839	37845	37845	37853	37853	37857	37857	37868
37868	37878	37878	37882	37882	37914	37914	37958	37958	37962	37962	37965
37965	37969	37969	37973	37973	37977	37977	38011	38011	38065	38065	38084
38084	38092	38092	38095	38095	38106	38106	38109	38109	38113	38113	38152
38152	38157	38157	38174	38174	38202	38202	38206	38206	38214	38214	38217
38217	38222	38222	38225	38225	38229	38229	38272	38272	38279	38279	38283
38283	38346	38346	38351	38351	38370	38370	38379	38379	38385	38385	38390
38390	38405	38405	38419	38419	38428	38428	38434	38434	38444	38444	38473
38473	38529	38541	38547	38547	38552	38552	38556	38563	38563	38572	38572
38579	38579	38610	38610	38627	38627	38632	38632	38638	38638	38644	38644
38655	38655	38699	38699	38706	38706	38712	38712	38717	38717	38722	38722
38730	38730	38740	38740	38746	38746	38752	38752	38757	38757	38767	38767
38807	38807	38834	38852	38852	38856	38856	38918	38918	38928	38928	39039
39039	39120	39120	39158	39158	39210	39210	39269	39269	39276	39276	39376
39376	39386	39386	39430	39430	39469	39469	39485	39485	39726	39726	39740
39740	39760	39760	39806	39806	39813	39813	39834	39834	39925	39925	39932
39932	39949	39949	39960	39960	40087	40087	40091	40091	40150	40150	40154
40154	40169	40169	40173	40173	40180	40180	40191	40191	40195	40195	40218
40218	40226	40226	40232	40232	40237	40237	40241	40241	40303	40303	40309
40309	40324	40324	40327	40327	40408	40408	40432	40432	40436	40436	40484
40489	40494	40498	40498	40503	40503	40509	40509	40515	40515	40528	40528
40542	40542	40548	40548	40552	40552	40561	40561	40572	40572	40577	40577
40590	40590	40594	40594	40597	40597	40605	40605	40622	40626	40626	40647
40647	40651	40651	40858	41032	41032	41070	41070	41210	41210	41228	41228
41310	41310	41337	41337	41351	41355	41355	41362	41367	41367	41385	41429
41429	41443	41561	41643	41643	41643	41648	41648	41651	41651	41664	41664
41664	41669	41669	41673	41673	41679	41679	41683	41683	41693	41693	41693
41698	41698	41698	41703	41703	41703	41954	41959	41964	41986	42041	42045

	42045	42053	42053	42057	42057	42061	42061	42087	42087	42099	42103	42109
	42114	42114	42129	42138	42171	42176	42176	42486	42486	42508	42662	42662
	42670	42674	42674	42683	42695	42695	42789	42789	42939	42939	43080	43080
	43192	43192	44751	44751	45169	45169	45604	45604	45674	45674	45786	45786
	46148	46148	46152	46152	46156	46156	46162	46162	46166	46166	46179	46179
	46183	46183	46201	46201	46219	46219	46243	46243	46251	46251	46280	46280
	46305	46305	46313	46313	46344	46344	46353	46353	46368	46368	46387	46387
	46391	46391	46403	46403	46407	46407	46416	46416	46428	46428	46432	46432
	46441	46441	46453	46453	46457	46457	46475	46475	46478	46478	46488	46488
	46492	46492	46544	46544	46569	46569	46576	46576	46583	46583	46590	46590
	46595	46595	46601	46601	46608	46608	46615	46615	46622	46622	46629	46629
	46635	46635	46642	46642	46648	46648	46655	46655	46662	46662	46668	46668
	46675	46675	46682	46682	46689	46689	46696	46696	46716	46716	46725	46725
	46731	46731	46740	46740	46744	46744	46749	46749	46763	46763	46781	46781
	46807	46807	46810	46810	46815	46815	46819	46819	46827	46827	46832	46832
	46875	46875	46886	46886	46893	46893	46895	46895	46908	46911	46911	46922
	46922	46926	46926	46947	46947	46952	46952	46990	46993	46993	47000	47003
	47003	47012	47012	47022	47022	47062	47062	47070	47070	47091	47091	47094
	47094	47098	47098	47101	47101	47109	47109	47112	47112	47136	47136	47139
	47139	47154	47154	47158	47158	47162	47162	47165	47165	47169	47169	47173
	47173	47176	47176	47181	47181	47187	47187	47190	47190	47197	47197	47201
	47201	47208	47208	47211	47211	47215	47215	47219	47219	47222	47222	47227
	47227	47230	47230	47237	47237	47241	47241	47248	47248	47251	47251	47255
	47255	47259	47259	47262	47262	47265	47265	47271	47271	47281	47281	47295
	47295	47299	47299	47302	47302	47338	47338	47338	47356	47356	47371	47371
	47371	47378	47378	47393	47393	47402	47402	47406	47406	47412	47412	47415
	47415	47423	47423	47431	47431	47470	47470	47474	47474	47478	47478	47495
	47495	47500	47500	47505	47505	47510	47510	47515	47515	47520	47520	47541
	47541	47546	47546	47551	47551	47561	47561	47566	47570	47575	47575	47647
	47647	47651	47651	47655	47655	47661	47661	47740	47740	47752	47759	47759
	47782	47782	47805	47805	47810	47810	47899	47899	47904	47904	47937	47937
	47943	47943	47947	47947	47984	47984	48033	48033	48084	48084	48087	48087
	48090	48090	48093	48093	48096	48096	48099	48099				
CCBR1.INT-TS	2823 #	5328	5328	5335	5335	5358	5358	5366	5366	5385	5385	5399
	5399	5403	5403	5409	5409	5455	5455	5459	5459	5463	5463	5472
	5472	5477	5477	5488	5488	5496	5496	5500	5500	5504	5504	5508
	5508	5515	5515	5519	5519	5523	5523	5531	5531	5535	5535	5540
	5540	5560	5560	5583	5583	5591	5591	5601	5601	5605	5605	5614
	5614	5617	5617	5626	5626	5630	5630	5659	5659	5662	5662	5676
	5676	5682	5682	5687	5687	5699	5699	5712	5712	5716	5716	5723
	5723	5733	5733	5750	5750	5785	5785	5796	5796	5824	5824	5899
	5899	5913	5913	5920	5920	5934	5934	5940	5940	5943	5943	5951
	5951	5954	5954	5957	5957	5963	5963	5968	5968	5978	5978	5992
	5992	5998	5998	6007	6007	6128	6128	6132	6132	6142	6142	6149
	6149	6156	6156	6160	6160	6171	6171	6174	6174	6221	6221	6225
	6225	6229	6229	6233	6233	6236	6236	6240	6240	6244	6244	6248
	6248	6256	6256	6265	6265	6270	6270	6274	6274	6281	6281	6285
	6285	6289	6289	6293	6293	6299	6299	6303	6303	6307	6307	6313
	6313	6317	6317	6321	6321	6327	6327	6331	6331	6335	6335	6341
	6341	6345	6345	6349	6349	6355	6355	6359	6359	6363	6363	6369
	6369	6373	6373	6377	6377	6383	6383	6387	6387	6391	6391	6397
	6397	6401	6401	6405	6405	6411	6411	6415	6415	6419	6419	6425
	6425	6433	6433	6464	6464	6471	6471	6478	6478	6487	6487	6503
	6503	6508	6508	6528	6528	6536	6536	6542	6542	6545	6545	6557
	6557	6613	6613	6662	6662	6667	6667	6695	6709	6709	6753	6753

Cross Reference Listing - Field Names and Defined Values

6803	6803	6825	6825	6832	6832	6858	6858	6932	6932	6965	6965
6978	6978	7014	7014	7033	7037	7037	7042	7042	7048	7048	7051
7051	7055	7055	7059	7059	7062	7062	7066	7066	7071	7071	7074
7074	7078	7078	7083	7083	7086	7086	7090	7090	7095	7095	7098
7098	7102	7102	7107	7107	7110	7110	7166	7166	7170	7170	7173
7173	7177	7177	7184	7184	7187	7187	7231	7231	7235	7235	7264
7264	7268	7268	7276	7276	7280	7280	7284	7284	7310	7310	7314
7314	7322	7322	7330	7330	7338	7338	7345	7345	7353	7353	7361
7361	7369	7369	7373	7373	7378	7378	7383	7383	7405	7405	7410
7410	7420	7420	7433	7433	7438	7438	7454	7454	7459	7459	7466
7466	7476	7476	7482	7482	7499	7499	7504	7504	7521	7521	7526
7526	7533	7533	7543	7543	7548	7548	7557	7557	7561	7561	7565
7565	7569	7569	7578	7578	7581	7581	7591	7591	7595	7595	7615
7615	7627	7627	7635	7635	7644	7644	7648	7648	7652	7652	7664
7664	7676	7676	7690	7690	7706	7706	7714	7714	7718	7718	7729
7729	7741	7741	7746	7746	7750	7750	7753	7753	7757	7757	7762
7762	7769	7769	7789	7789	7809	7809	7829	7829	7833	7833	7850
7850	7861	7861	7873	7873	7908	7908	7912	7912	7915	7915	7944
7944	7949	7949	7958	7958	7963	7963	7975	7975	7986	7986	8009
8009	8014	8014	8026	8026	8032	8032	8058	8058	8063	8063	8068
8068	8072	8072	8076	8076	8080	8080	8094	8094	8143	8143	8187
8187	8191	8191	8199	8199	8206	8206	8910	8927	8950	9027	9039
9315	9324	9396	9396	9401	9401	9416	9420	9420	9425	9430	9430
9441	9449	9449	9459	9459	9559	9559	9570	9570	9670	9670	9690
9690	9694	9770	9770	9778	9778	9830	10817	10817	10825	10825	10868
10868	10879	10879	10926	10926	10930	10930	10934	10934	11029	11029	11034
11034	11149	11149	11154	11154	11329	11329	11337	11340	11340	11353	11499
11712	11771	11771	11782	11782	11799	11799	11813	11813	11885	11885	12035
12035	12042	12042	12224	12238	12238	12333	12333	12343	12343	12372	12372
12385	12385	12400	12400	12513	12513	12519	12519	12550	12550	12556	12562
12562	12580	12580	12594	12594	12621	12621	12633	12637	12637	12642	12642
12700	12700	12706	12706	12712	12712	12716	12716	12721	12721	12726	12726
12731	12731	12734	12734	12858	12858	12878	12878	13085	13085	13469	13480
13480	13588	13588	13625	13625	13670	13670	13684	13684	13702	13702	13878
13878	14077	14088	14088	14108	14113	14113	14118	14118	14426	14426	14442
14442	14446	14446	14471	14475	14501	14501	14505	14505	14666	14666	14690
14690	14702	14702	14713	14713	14735	14735	14837	14837	14843	14843	14973
14973	15008	15008	15016	15016	15073	15073	15079	15079	15084	15084	15088
15088	15095	15095	15098	15098	15123	15123	15166	15166	15330	15335	15386
15390	15390	15418	15418	15422	15422	15426	15426	15690	15690	15711	15711
15742	15742	15777	15777	15812	15812	15830	15830	15847	15847	15876	15876
15880	15880	15889	15905	15905	15918	15997	15997	16060	16060	16064	16064
16069	16069	16114	16114	16120	16120	16137	16137	16141	16141	16200	16200
16213	16230	16230	16290	16290	16435	16435	16483	16483	16525	16525	16563
16563	16574	16574	16599	16599	16604	16604	16624	16624	16629	16668	16668
16991	16991	16997	17033	17055	17118	17118	17123	17123	17160	17160	17169
17169	17173	17173	17177	17177	17238	17238	17246	17305	17474	17474	17494
17562	17562	17566	17566	17592	17592	17610	17610	17634	17634	19042	19042
19053	19053	19065	19065	19077	19077	19154	19154	19167	19167	19237	19237
19242	19242	19304	19311	19316	19349	19354	19359	19400	19405	19470	19470
19479	19479	19484	19484	19583	19583	20124	20124	20129	20129	20135	20135
20140	20140	20156	20156	20156	20177	20177	20200	20200	20211	20211	20244
20244	20253	20253	20262	20262	20270	20270	20279	20279	20288	20288	20297
20297	20305	20305	20314	20314	20323	20323	20332	20332	20340	20340	20350
20350	20355	20355	20360	20360	20365	20365	20370	20370	20375	20375	20379

Cross Reference Listing - Field Names and Defined Values

20379	20384	20384	20389	20389	20394	20394	20399	20399	20404	20404	20409
20409	20429	20429	20438	20438	20450	20450	20455	20455	20465	20465	20469
20469	20491	20491	20495	20495	20499	20499	20536	20536	20540	20540	20545
20545	20548	20548	20553	20553	20573	20573	20608	20608	20614	20614	20618
20618	20623	20623	20627	20627	20633	20633	20637	20637	20643	20643	20647
20647	20653	20653	20657	20657	20663	20663	20667	20667	20672	20672	20676
20676	20682	20682	20686	20686	20692	20692	20696	20696	20702	20702	20706
20706	20712	20712	20716	20716	20721	20721	20742	20742	20751	20751	20780
20780	20788	20788	20792	20792	20806	20806	20811	20811	20901	20901	20914
20914	20930	20930	20950	20955	20955	20962	20962	20966	20966	20972	20972
20976	20976	21032	21046	21051	21102	21102	21117	21117	21137	21137	21144
21166	21166	21206	21206	21213	21213	21263	21263	21320	21320	21324	21324
21327	21327	21350	21386	21386	21396	21396	21412	21439	21439	21454	21454
21468	21468	21472	21472	21543	21543	21546	21546	21604	21604	21615	21615
21621	21621	21655	21655	21665	21665	21698	21698	21730	21730	21758	21758
21787	21787	21794	21794	21808	21808	21848	21848	21858	21858	21868	21868
21874	21874	21889	21889	21903	21903	21912	21912	21952	21952	21969	21969
21979	21979	21999	21999	22004	22004	22012	22012	22021	22021	22429	22429
22429	22482	22482	22486	22486	22546	22547	22547	22638	22638	22677	22677
22785	22785	23001	23001	23004	23004	23031	23031	23035	23035	23039	23039
23050	23050	23079	23079	23096	23096	23107	23107	23129	23129	23168	23168
23172	23172	23176	23176	23187	23187	23205	23205	23237	23237	23245	23245
23279	23279	23333	23333	23437	23437	23469	23479	23688	23688	23694	23761
23761	23808	23808	23818	23818	23922	23922	23947	23947	23957	23957	24015
24015	24036	24062	24062	24103	24103	24125	24251	24251	24314	24337	24343
24367	24367	24423	24423	24457	24463	24463	24472	24472	24477	24477	24486
24495	24495	24526	24526	24541	24541	24647	24647	24665	24665	24669	24669
24680	24680	24684	24684	24708	24708	24760	24760	24782	24782	24797	24797
24802	24802	24805	24805	24835	24835	24869	24869	24930	24930	24933	24933
24952	24952	24959	24959	25025	25025	25041	25041	25047	25047	25051	25051
25092	25092	25110	25110	25119	25119	25165	25165	25204	25204	25243	25243
25284	25284	25292	25292	25299	25299	25368	25368	25612	25612	25634	25634
25649	25650	25650	25677	25677	25681	25681	25714	25714	25791	25792	25792
25797	25797	25973	25973	25994	25994	26003	26003	26006	26009	26009	26009
26017	26017	26029	26029	26104	26111	26111	26143	26143	26146	26146	26172
26212	26212	26293	26293	26361	26361	26364	26364	26370	26373	26373	26373
26444	26465	26465	26501	26501	26504	26504	26582	26583	26790	26790	26790
26794	26794	26848	26848	26853	26853	26896	26896	27014	27017	27017	27017
27023	27023	27026	27026	27044	27044	27051	27051	27063	27067	27067	27067
27075	27102	27102	27292	27304	27333	27333	27375	27375	27379	27379	27441
27450	27665	27665	27687	27687	27732	27732	27775	27775	27779	27797	27797
27797	27801	27801	27817	27817	27827	27827	27835	27835	27839	27839	27845
27845	27849	27849	27855	27855	27859	27859	27868	27868	27872	27872	27876
27876	27911	27911	27916	27916	27938	27964	27984	27984	27995	28008	28008
28027	28027	28030	28049	28049	28065	28065	28075	28075	28106	28106	28117
28117	28144	28144	28147	28147	28155	28155	28166	28178	28178	28183	28183
28188	28188	28213	28213	28245	28245	28249	28264	28264	28268	28268	28279
28285	28285	28298	28298	28312	28312	28324	28328	28328	28334	28334	28338
28338	28341	28341	28418	28418	28418	28428	28428	28453	28453	28477	28477
28494	28494	28500	28500	28506	28506	28534	28534	28538	28538	28563	28563
28584	28589	28598	28598	28601	28601	28619	28619	28661	28661	28665	28665
28670	28670	28746	28770	28770	28774	28774	28779	28779	28783	28783	28793
28793	28797	28797	28824	28824	28828	28828	28836	28836	28855	28855	29074
29074	29089	29089	29112	29112	29116	29116	29134	29134	29138	29138	29142
29142	29176	29176	29246	29246	29250	29250	29258	29269	29269	29381	29381

29488	29488	29491	29491	29511	29511	29530	29535	29535	29538	29538	29560
29560	29567	29567	29585	29585	29590	29590	29595	29595	29600	29600	29605
29605	29624	29636	29636	29660	29660	29670	29670	29680	29680	29684	29684
29689	29698	29698	29712	29712	29717	29717	29723	29723	29727	29727	29731
29818	29818	29822	29822	29828	29828	29836	29836	29843	29843	29850	29850
29869	29869	29877	29877	29912	29912	29986	30029	30105	30105	30202	30202
30226	30226	30244	30244	30249	30249	30253	30253	30257	30257	30267	30267
30273	30273	30277	30277	30281	30281	30286	30286	30296	30296	30352	30352
30374	30374	30380	30380	30388	30388	30451	30451	30457	30457	30461	30461
30481	30481	30492	30492	30499	30499	30518	30518	30532	30532	30575	30581
30586	30586	30601	30601	30611	30611	30620	30620	30626	30626	30632	30632
30636	30636	30641	30641	30714	30714	30725	30735	30735	30756	30756	30760
30760	30781	30781	30787	30787	30830	30875	30875	30879	30879	30888	30888
30909	30909	30926	30926	30987	30987	30998	30998	31002	31002	31009	31009
31014	31014	31021	31021	31026	31026	31049	31049	31062	31062	31112	31112
31129	31227	31227	31230	31230	31241	31241	31349	31349	31436	31442	31442
31456	31456	31460	31495	31498	31498	31505	31505	31514	31514	31588	31588
31599	31630	31650	31650	31688	31688	31704	31704	31708	31708	31745	31745
31926	31926	32044	32044	32081	32082	32082	32100	32100	32123	32124	32124
32131	32131	32163	32163	32172	32172	32372	32372	32379	32379	32385	32385
32396	32396	32400	32400	32404	32404	32420	32420	32423	32423	32426	32426
32438	32438	32446	32459	32459	32467	32478	32478	32482	32482	32494	32494
32502	32511	32511	32525	32545	32545	32564	32564	32567	32567	32570	32570
32574	32574	32591	32591	32595	32595	32599	32599	32603	32603	32607	32607
32611	32611	32615	32615	32625	32625	32629	32629	32635	32635	32639	32639
32645	32645	32649	32649	32655	32655	32659	32659	32664	32664	32667	32667
32671	32671	32687	32687	32692	32702	32702	32707	32707	32733	32747	32747
32791	32791	32794	32815	32815	32823	32823	32829	32829	32833	32833	32837
32837	32862	32862	32880	32880	32884	32884	32903	32903	32916	32916	32920
32920	32938	32938	32952	32952	32984	32984	32992	32992	32996	32996	33019
33019	33027	33027	33037	33037	33051	33051	33061	33061	33065	33065	33128
33128	33150	33150	33467	33467	33484	33484	33527	33527	33535	33535	33538
33538	33542	33542	33545	33545	33559	33572	33575	33575	33579	33579	33611
33611	33615	33615	33619	33619	33629	33629	33639	33639	33650	33650	33650
33654	33654	33660	33660	33669	33669	33673	33673	33677	33677	33820	33820
33831	33831	33904	33904	33917	33917	33956	33956	33999	33999	34005	34005
34029	34029	34033	34033	34072	34072	34076	34076	34089	34089	34093	34093
34133	34133	34147	34147	34150	34150	34154	34154	34157	34157	34168	34168
34172	34172	34178	34178	34198	34198	34241	34241	34260	34260	34264	34264
34268	34268	34275	34275	34294	34294	34301	34301	34319	34319	34351	34363
34363	34373	34380	34380	34388	34394	34394	34405	34405	34441	34441	34458
34458	34498	34498	34505	34505	34561	34561	34570	34570	34610	34610	34658
34658	34679	34679	34682	34682	34701	34701	34704	34707	34707	34714	34714
34838	34838	34864	34864	34891	34891	34900	34900	34904	34904	34908	34908
34913	34913	35014	35014	35031	35031	35064	35064	35067	35067	35079	35079
35083	35083	35091	35091	35107	35107	35111	35111	35115	35115	35119	35119
35123	35123	35127	35127	35176	35176	35186	35186	35205	35205	35252	35252
35268	35268	35272	35288	35288	35315	35315	35328	35328	35331	35331	35363
35369	35369	35372	35372	35376	35376	35382	35382	35385	35385	35390	35390
35394	35394	35402	35402	35405	35405	35408	35408	35412	35412	35418	35418
35437	35437	35456	35456	35460	35460	35469	35469	35478	35478	35482	35482
35489	35489	35493	35493	35498	35498	35503	35503	35512	35512	35516	35516
35522	35522	35526	35526	35532	35532	35536	35536	35567	35567	35571	35571
35576	35576	35585	35585	35590	35590	35599	35599	35640	35651	35660	35670
35670	35698	35698	35703	35703	35741	35741	35779	35779	35779	35783	35783

Cross Reference Listing - Field Names and Defined Values

35798	35798	35802	35802	35814	35814	35818	35818	35822	35822	35826	35826
35834	35834	35838	35838	35842	35842	35850	35850	35853	35853	35857	35857
35861	35861	35873	35873	35897	35897	35905	35905	35914	35914	35919	35919
35923	35923	35927	35927	35931	35931	35935	35935	35939	35939	35987	35987
36000	36000	36004	36004	36007	36007	36047	36047	36053	36053	36060	36060
36063	36063	36083	36083	36088	36088	36092	36092	36097	36097	36101	36101
36106	36106	36112	36112	36116	36116	36121	36121	36136	36136	36139	36139
36152	36152	36156	36156	36184	36184	36228	36228	36233	36233	36238	36238
36242	36242	36248	36248	36248	36251	36251	36289	36289	36304	36304	36312
36316	36316	36337	36337	36345	36350	36354	36354	36479	36479	36496	36496
36563	36563	36580	36580	36615	36615	36619	36619	36619	36627	36627	36667
36667	36679	36679	36684	36684	36684	36701	36701	36715	36715	36732	36732
36745	36745	36755	36755	36782	36782	36796	36799	36799	36809	36809	36841
36841	36849	36849	36853	36853	36871	36871	36882	36882	36896	36905	36905
36908	36908	36916	36916	37099	37099	37105	37105	37110	37110	37115	37115
37121	37121	37131	37131	37137	37137	37182	37189	37196	37196	37205	37205
37213	37219	37248	37248	37252	37252	37256	37256	37271	37271	37275	37275
37279	37279	37283	37283	37312	37312	37317	37317	37322	37322	37336	37375
37375	37380	37380	37384	37384	37388	37388	37400	37400	37428	37428	37432
37432	37437	37437	37441	37441	37488	37488	37537	37537	37560	37560	37563
37563	37586	37586	37629	37629	37635	37635	37641	37641	37647	37647	37663
37663	37672	37672	37686	37686	37694	37694	37698	37698	37719	37719	37723
37723	37727	37727	37737	37737	37744	37744	37770	37770	37789	37789	37801
37801	37807	37807	37813	37813	37831	37831	37839	37839	37845	37845	37853
37853	37857	37857	37868	37868	37878	37878	37882	37882	37914	37914	37958
37958	37962	37962	37965	37965	37969	37969	37973	37973	37977	37977	38011
38011	38065	38065	38084	38084	38092	38092	38095	38095	38106	38106	38109
38109	38113	38113	38152	38152	38157	38157	38174	38174	38202	38202	38206
38206	38214	38214	38217	38217	38222	38222	38225	38225	38229	38229	38272
38272	38279	38279	38283	38283	38346	38346	38351	38351	38370	38370	38379
38379	38385	38385	38390	38390	38405	38405	38419	38419	38428	38428	38434
38434	38444	38444	38473	38473	38529	38541	38547	38547	38552	38552	38556
38563	38563	38572	38572	38579	38579	38610	38610	38627	38627	38632	38632
38638	38638	38644	38644	38655	38655	38699	38699	38706	38706	38712	38712
38717	38717	38722	38722	38730	38730	38740	38740	38746	38746	38752	38752
38757	38757	38767	38767	38807	38807	38834	38852	38852	38856	38856	38918
38918	38928	38928	39039	39039	39120	39120	39158	39158	39210	39210	39269
39269	39276	39276	39376	39376	39386	39386	39430	39430	39469	39469	39485
39485	39726	39726	39740	39740	39760	39760	39806	39806	39813	39813	39834
39834	39925	39925	39932	39932	39949	39949	39960	39960	40087	40087	40091
40091	40150	40150	40154	40154	40169	40169	40173	40173	40180	40180	40191
40191	40195	40195	40218	40218	40226	40226	40232	40232	40237	40237	40241
40241	40303	40303	40309	40309	40324	40324	40327	40327	40408	40408	40432
40432	40436	40436	40484	40489	40494	40498	40498	40503	40503	40509	40509
40515	40515	40528	40528	40542	40542	40548	40548	40552	40552	40561	40561
40572	40572	40577	40577	40590	40590	40594	40594	40597	40597	40605	40605
40622	40626	40626	40647	40647	40651	40651	40858	41032	41032	41070	41070
41210	41210	41228	41228	41310	41310	41337	41337	41351	41355	41355	41362
41367	41367	41385	41429	41429	41443	41561	41643	41643	41643	41648	41648
41651	41651	41664	41664	41664	41669	41669	41673	41673	41679	41679	41683
41683	41693	41693	41693	41698	41698	41698	41703	41703	41703	41954	41959
41964	41986	42041	42045	42045	42053	42053	42057	42057	42061	42061	42087
42087	42099	42103	42109	42114	42114	42129	42138	42171	42176	42176	42486
42486	42508	42662	42662	42670	42674	42674	42683	42695	42695	42789	42789
42939	42939	43080	43080	43192	43192	44751	44751	45169	45169	45604	45604

; Cross Reference Listing - Field Names and Defined Values

	45674	45674	45786	45786	46148	46148	46152	46152	46156	46156	46162	46162
	46166	46166	46179	46179	46183	46183	46201	46201	46219	46219	46243	46243
	46251	46251	46280	46280	46305	46305	46313	46313	46344	46344	46353	46353
	46368	46368	46387	46387	46391	46391	46403	46403	46407	46407	46416	46416
	46428	46428	46432	46432	46441	46441	46453	46453	46457	46457	46475	46475
	46478	46478	46488	46488	46492	46492	46544	46544	46569	46569	46576	46576
	46583	46583	46590	46590	46595	46595	46601	46601	46608	46608	46615	46615
	46622	46622	46629	46629	46635	46635	46642	46642	46648	46648	46655	46655
	46662	46662	46668	46668	46675	46675	46682	46682	46689	46689	46696	46696
	46716	46716	46725	46725	46731	46731	46740	46740	46744	46744	46749	46749
	46763	46763	46781	46781	46807	46807	46810	46810	46815	46815	46819	46819
	46827	46827	46832	46832	46875	46875	46886	46886	46893	46893	46895	46895
	46908	46911	46911	46922	46922	46926	46926	46947	46947	46952	46952	46990
	46993	46993	47000	47003	47003	47012	47012	47022	47022	47062	47062	47070
	47070	47091	47091	47094	47094	47098	47098	47101	47101	47109	47109	47112
	47112	47136	47136	47139	47139	47154	47154	47158	47158	47162	47162	47165
	47165	47169	47169	47173	47173	47176	47176	47181	47181	47187	47187	47190
	47190	47197	47197	47201	47201	47208	47208	47211	47211	47215	47215	47219
	47219	47222	47222	47227	47227	47230	47230	47237	47237	47241	47241	47248
	47248	47251	47251	47255	47255	47259	47259	47262	47262	47265	47265	47271
	47271	47281	47281	47295	47295	47299	47299	47302	47302	47338	47338	47338
	47356	47356	47371	47371	47371	47378	47378	47393	47393	47402	47402	47406
	47406	47412	47412	47415	47415	47423	47423	47431	47431	47470	47470	47474
	47474	47478	47478	47495	47495	47500	47500	47505	47505	47510	47510	47515
	47515	47520	47520	47541	47541	47546	47546	47551	47551	47561	47561	47566
	47570	47575	47575	47647	47647	47651	47651	47655	47655	47661	47661	47740
	47740	47752	47759	47759	47782	47782	47805	47805	47810	47810	47899	47899
	47904	47904	47937	47937	47943	47943	47947	47947	47984	47984	48033	48033
	48084	48084	48087	48087	48090	48090	48093	48093	48096	48096	48099	48099
CM.ODD.ADD	2793 #	42744	42753	42758	42768	42778	42793	42798	42865	42892	42901	42906
	42917	42927	42943	42948	43006	43033	43042	43047	43058	43068	43104	43158
	43226	43231	43236									
DBZ.SC	2815 #	7042	7982	9191	9230	9591	9625	12851	13080	14208	14219	14234
	14254	14462	14480	14672	15782	17540	21268	26330	28485	29508	29515	29843
	29912	30600	30610	30614	30619	30631	30714	30786	35437	35697	36706	36866
	38971	38975	46254	46316	46411	46436	46461	46495	46768	47386	47983	
DSIZE	2814 #	9149	9154	9159	9164							
F1.XOR23	2840 #	12864	13090	13884	14436	14828	14980	15776	16583	16623	27342	29749
	30379											
FLAG0	2832 #	5492	5643	6551	6652	6670	6682	6978	7610	7630	7639	7679
	7683	7690	9195	9234	9596	9630	11012	11021	11131	11141	11328	11447
	11563	11663	11682	11690	11794	15114	15321	15495	15899	16667	18762	20445
	22935	22964	22980	22994	24508	24984	26306	26333	26349	27911	28489	28528
	28750	29903	30240	32536	32836	32844	32848	32889	32892	32907	32964	32969
	33031	33040	33716	33802	34154	34181	34264	34275	34280	34291	37369	38011
	38487	38664	39055	39453								
FLAG1	2833 #	7552	9498	11923	12845	13064	13324	13331	13982	14425	15806	15812
	15856	15861	15869	16927	17582	20147	20165	20171	20561	20566	23064	23338
	23353	23674	23876	25709	25957	29693	29735	29953	29968	29973	30114	30118
	30281	30374	30492	35703	37322	37375	39007	39098	39105			
FLAG1TOO	2839 #	12337	12379	12405	12694	13610	14494	14763	15520	17233	18402	18406
	26671	26675	26679	26683	39062	41452	42045	43168	43191	43215	46543	46547
FLAG2	2834 #	5967	7968	8051	9553	9559	9565	9569	9580	11489	11651	11671
	11678	14790	14799	15079	15095	15157	18424	18431	18483	18487	25173	25686
	30277	30680	31301	31549	31575	31578	33806	34561	37353	42204	47540	47545

Cross Reference Listing - Field Names and Defined Values

	24789	24844	24860	24873	25123	25376	25388	25969	25973	25994	26006	26009
	26014	26017	26021	26029	26032	26189	26198	26306	26333	26361	26364	26370
	26373	27724	27728	28849	28866	29228	29238	29339	29370	29754	29758	30019
	30025	31126	31245	31286	31291	31301	31305	32037	32041	33556	33575	33956
	34557	34566	34616	34673	35451	35507	35543	35547	35580	35593	35617	35645
	35655	35948	36097	36121	36527	36715	36813	36841	36942	37256	37346	37380
	37388	37690	37709	37733	37755	37776	37804	37807	37813	37819	37827	37841
	37857	37887	37918	38035	38106	38146	38149	38160	38362	38449	38616	38622
	38807	38811	39205	39210	39245	39264	39269	39295	39434	39457	39463	39475
	39644	39755	39944	39955	40256	40306	40316	40390	40512	40522	40991	40996
	41001	41027	41032	41038	41065	41070	41076	41103	41108	41114	41140	41145
	41150	41176	41181	41186	41233	41260	41265	41271	41333	41359	41367	41381
	41425	41486	41490	41494	41505	41525	41529	41533	41544	41600	41608	41614
	41728	41777	41782	41807	41821	41826	41831	41860	41869	41892	41901	41923
	41932	42049	42053	42057	42061	42087	42176	42217	42249	42259	42276	42282
	42292	42315	42324	42336	42346	42352	42357	42398	42428	42435	42439	42443
	42448	42465	42479	42483	42493	42503	42512	42516	42626	42645	42655	42678
	42689	42799	42855	42949	43001	43080	43088	43192	43204	44364	44374	44393
	44489	44505	44585	44601	44675	44691	44751	44763	44779	44842	44861	44951
	44970	45002	45080	45098	45194	45217	45297	45315	45370	45380	45399	45457
	45467	45486	45604	45613	45632	45723	45742	45839	45858	45916	45928	45950
	46030	46049	46159	46243	46251	46305	46313	46391	46407	46416	46432	46441
	46457	46475	46492	46569	46576	46583	46590	46595	46601	46608	46615	46622
	46629	46635	46642	46648	46655	46662	46668	46675	46682	46689	46696	46744
	46886	46893	46895	46903	46911	46915	46922	46926	46947	46993	47003	47012
	47022	47066	47094	47101	47112	47154	47165	47176	47184	47190	47197	47211
	47222	47230	47237	47251	47262	47268	47274	47278	47284	47295	47302	47356
	47378	47419	47556									
IRD1TST	2781 #	5358	5422	5455	5459	5463	5523	5605	5667	5676	5882	6122
	6139	6145	6277	6437	6653	6657	6702	6769	6969	6987	7021	7027
	7042	7045	7162	7190	7389	7398	7416	7426	9170	9175	9180	9185
	9191	9195	9199	9203	9209	9214	9219	9224	9230	9234	9238	9242
	9554	9565	9581	9591	9596	9600	9608	9614	9619	9625	9630	9634
	9642	9648	9653	12852	13081	14209	14220	14235	14255	14673	15340	15783
	16294	17470	17488	17578	17616	18868	18873	18878	18883	18888	18893	18898
	18903	18915	18927	18931	18935	18939	18943	18947	18951	18955	18971	18976
	18981	18986	18991	18996	19001	19005	19022	19098	19189	19250	19322	19410
	19414	19418	19461	19465	19474	19493	19566	19583	19607	19611	19615	19619
	19623	19627	19631	19635	19659	19664	19669	19674	19679	19684	19689	19693
	19719	20458	20594	20914	20962	22458	22627	22643	22674	22722	22789	22994
	23050	23053	23107	23187	23450	23454	23527	23551	23761	23808	23818	23877
	23928	24015	24103	24251	24318	24328	24372	24427	24569	24712	24789	24844
	24860	24873	25123	25376	25388	26189	26198	26306	26333	27724	27728	28849
	28866	29228	29238	29339	29370	30025	31126	31245	31286	31291	31301	31305
	32037	32041	33556	33575	33956	34557	34616	34673	35451	35507	35543	35547
	35580	35593	35617	35948	36097	36121	36527	36715	36841	37256	37346	37380
	37388	37690	37709	37733	37755	37776	37804	37807	37813	37819	37827	37841
	37857	37918	38035	38106	38146	38149	38160	38362	38449	38616	38622	38807
	39205	39210	39245	39264	39269	39295	39434	39644	39755	39944	39955	40256
	40306	40316	40390	40512	40522	40996	41032	41070	41108	41181	41265	41490
	41529	41614	41728	41782	41807	41826	41869	41901	42217	42259	42324	42336
	42346	42357	42398	42428	42435	42439	42443	42465	42479	42493	42503	42626
	42645	42655	42799	42855	42949	43001	43080	43088	43192	43204	44364	44374
	44393	44489	44505	44585	44601	44675	44691	44751	44763	44779	44842	44861
	44951	44970	45002	45080	45098	45194	45217	45297	45315	45370	45380	45399

		45457	45467	45486	45604	45613	45632	45723	45742	45839	45858	45916	45928
		45950	46030	46049	46159	46243	46251	46305	46313	46391	46407	46416	46432
		46441	46457	46475	46492	46569	46576	46583	46590	46595	46601	46608	46615
		46622	46629	46635	46642	46648	46655	46662	46668	46675	46682	46689	46696
		46744	46886	46893	46895	46898	46903	46911	46915	46922	46926	46947	46993
		47003	47012	47022	47066	47073	47094	47101	47112	47154	47165	47176	47184
		47190	47197	47211	47222	47230	47237	47251	47262	47268	47274	47278	47284
		47295	47302	47356	47378	47419							
IRDX	2782 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219	
		9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
		9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
		14235	14255	14673	15783	42192	42196	42200	42736	42815	42821	42840	42847
		42883	42967	42986	42993	43024	43029	43109	43126	43132	43140	43243	43252
		43261	44349	44364	44416	44426	44432	44438	44593	44597	44619	44625	44630
		44709	44714	44821	44884	44891	44898	44965	44992	45012	45022	45027	45033
		45040	45058	45075	45094	45121	45126	45131	45138	45149	45169	45187	45194
		45201	45207	45212	45252	45276	45310	45356	45420	45425	45517	45548	45659
		45786	45881	45916	45978	45991							
LOD.BRA	2786 #	5523	5605	6145	6277	6653	6769	6987	7021	7162	7389	7398	
		7416	7426	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214
		9219	9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600
		9608	9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209
		14220	14235	14255	14673	15783	35451	35507	35543	35547	35580	35948	36097
		36121	37690	37709	37733	37804	37807	37813	37819	37827	37841	37857	37918
		38035	38146	38149	38160	38362	38449	39238	39287	39944	39955	40306	40316
		40512	40522	42799	42949	43088	43204	44374	44393	44410	44489	44505	44585
		44601	44675	44691	44763	44779	44842	44861	44878	44951	44970	44986	45080
		45098	45115	45194	45217	45235	45297	45315	45332	45380	45399	45416	45467
		45486	45503	45613	45632	45649	45723	45742	45759	45839	45858	45875	45928
		45950	45969	46030	46049	46066	46159	47184	47190	47230	47268	47274	47278
		47284	47419										
LOD.INC.BRA	2785 #	5523	5605	6145	6277	6653	6769	6987	7021	7162	7389	7398	
		7416	7426	8333	8388	8905	8977	9019	9024	9094	9100	9170	9175
		9180	9185	9191	9195	9199	9203	9209	9214	9219	9224	9230	9234
		9238	9242	9271	9276	9281	9287	9292	9297	9306	9378	9396	9401
		9554	9565	9581	9591	9596	9600	9608	9614	9619	9625	9630	9634
		9642	9648	9653	9666	11115	11321	11557	11646	11902	12034	12041	12852
		13081	13159	13235	13240	13455	13767	13787	13792	14209	14220	14235	14255
		14310	14330	14335	14651	14661	14673	14868	14874	14941	14955	14972	15300
		15477	15783	15987	16429	16734	16747	16803	16817	16859	16901	16906	16964
		16969	17071	17312	17317	17334	17442	17450	17456	18388	18391	18436	18443
		18448	18455	18498	18502	18513	18517	18521	18524	18652	18672	18675	18679
		21006	21011	21016	21021	22417	22425	22497	22509	22515	22526	22557	22566
		22575	22582	22597	22604	22610	22653	22662	22698	22705	22711	25602	25607
		26089	26096	26101	26418	26424	26436	26440	26452	29458	29809	29920	30084
		30090	30095	30100	30175	30181	30186	30196	30325	30332	30337	30428	30434
		30439	31412	31422	31427	31431	31442	33475	33483	33488	35451	35507	35543
		35547	35580	35948	36097	36121	36167	36627	37690	37709	37733	37804	37807
		37813	37819	37827	37841	37857	37918	38035	38146	38149	38160	38362	38449
		39944	39955	40306	40316	40512	40522	42799	42949	43088	43204	44374	44393
		44489	44505	44585	44601	44675	44691	44763	44779	44842	44861	44951	44970
		45080	45098	45194	45217	45297	45315	45380	45399	45467	45486	45613	45632
		45723	45742	45839	45858	45928	45950	46030	46049	46159	47104	47184	47190
		47230	47268	47274	47278	47284	47419						
MM.ALLOW.INT	2821 #	23000	23060	23079	23145	23198	23296	23326	23436	23540	23562	23817	

Cross Reference Listing - Field Names and Defined Values

	23834	24196	34436	35521	35562	35589	39919	39975	40034	40045	40261	40376
	40386											
MM.NOINT	2836 #	7853	7857	12348	13591	15390	29650	29707	45602	47660	47937	48013
NO.FPA	2816 #	35864										
NOP	2779 #											
PSLC	2826 #	25620	28241	28253	32828	32832	32840	33019				
PSLTP	2827 #	39404	39408	39412	39416	39420						
REGMODE	2797 #	8405	8528	9045	9050	9383	9486	9775	11234	11239	11244	14716
	14744	14836	14842	14853	16771	16812	16827	16876	16911	16916	16956	16974
	16979	18364	18372	19382	19534	19566	19583					
RET.DINH	2784 #	5484	37782	40394								
RETURN	2783 #	5496	5508	5651	5676	5702	5737	5818	5882	5888	5974	5978
	6001	6459	7310	7557	7569	7618	7702	7710	7805	7824	7829	7833
	7864	7868	7877	7903	7949	8089	8154	8168	9199	9203	9238	9242
	9603	9614	9619	9637	9648	9653	10868	10875	10879	10992	10996	11002
	11380	12137	12142	12146	12162	12229	12233	12360	12365	12372	12542	12576
	12612	12628	12646	12700	12716	12721	12734	12738	12745	12892	12959	12964
	12969	12979	12985	12990	13320	13343	13348	13375	13596	13641	13665	13678
	13693	13712	13716	13867	13974	13987	13992	14098	14113	14259	14513	14520
	14795	15120	15147	15198	15232	15239	15243	15407	15418	15422	15426	15551
	15837	15842	16015	16053	16060	16064	16074	16294	16544	16792	16894	17076
	17102	17118	17123	17547	17557	17566	19150	20504	20509	20514	20519	20774
	21160	21331	21412	21483	21999	22004	22752	22765	22773	22789	22805	23974
	23997	24010	24024	24884	25388	25395	25406	27466	27540	29350	29428	29780
	30655	30659	30728	30898	30938	31052	31134	31337	31349	32214	32251	32288
	32420	32423	32426	32438	32459	32478	32494	32545	32574	32691	32738	32747
	33087	33093	33096	33120	33138	33150	33156	33176	33182	33190	33198	33208
	33221	33235	33244	34383	34394	34401	34409	34447	34451	34463	34873	34877
	34880	34896	34908	34913	34916	35706	35958	36512	36643	36818	36966	37361
	37437	37441	37513	37675	37679	37682	37686	37758	37792	37822	37918	38040
	38493	38840	38852	38888	38902	38956	38965	38999	39233	39504	39656	39693
	39760	39822	39828	39899	39903	39935	39939	39987	39991	40050	40066	40071
	40256	40270	40365	40498	40503	40522	40528	40542	40548	40552	40561	40572
	40590	40610	40626	40647	40651	40709	40723	40770	40782	40825	40866	40886
	40906	40910	40914	40934	40938	40942	44523	44534	44551	44805	45678	46926
	46955	47083	47284	47389	47431	47495	47500	47505	47510	47520	47647	47655
	47773	47815	47840	47852	47868	47882	47893	47904	47909	47963	47979	47997
	48001	48044										
SPASTA	2810 #	22098	34867	34956	37863	38762	38923	38962	39153	40986	41021	41059
	41097	41135	41171	41252	41481	41520	41595	41668	41749	41814	42188	42225
	42402	42579	42615	42715	42784	42803	42934	42954	43075	43093		
SRKSTA	2809 #	6483	6800	7113	7639	7655	7667	7686	7782	7786	7792	7812
	7895	7900	7920	9511	9676	9681	9750	9770	9805	11383	12603	15104
	15131	15206	17531	17547	18198	18217	18274	18286	18315	18330	18402	18410
	18414	18418	18656	18695	20216	20797	21311	27499	27513	27526	27534	28559
	28574	28725	28746	30356	30454	30747	30826	30915	30919	32200	32203	32206
	32211	32237	32240	32243	32248	32274	32277	32280	32285	33208	33530	33563
	35299	35433	35954	35958	36043	36069	36128	38953	38971	38975	40300	40484
	40489	40494	40506	40532	40538	40622	40630	41306	41315	41409	45058	47953
STACKFLG	2837 #	6233	37182	37189	37219	38910	39924	39931				
UVLTR	2828 #	20154	20156	20552	20553	21105	21108	21209	21210	21218	21218	21593
	21594	21674	21675	21689	21690	21797	21798	21884	21885	29207	29209	29619
	29620	34887	34952	35585	35599	35612	35625	35632	35632	35698	35703	35746
	36125	36268	36470	36470	36488	36488	36554	36554	36572	36572	36674	36679
	36772	36848	36892	37236	37241	37266	37288	37409	37810	37816	38259	38418

Cross Reference Listing - Field Names and Defined Values

		38419	38931	39172	39364	39381	39437	39796	39796	40028	40028	40042	40042
		40075	40080	40158	40163	40198	40201	40250	40365	40371	40371	40383	40383
		40421	40426	40441	40554	40557	40568	40577	40586	40594	40622	40630	40643
		40657	40657	40687	40687	40751	40751	40810	40810	40852	40854	47193	47204
		47233	47244										
WBUS0		2803 #	5381	5389	5393	5587	7132	8640	26130	26496	27423	27454	27677
		28258	29495	29500	29571	30606	30641	30832	30884	30893	31049	31482	31535
		31539	32085	35039	35074	35102	39478	41461	41490	41529	41614	41643	41664
		41693	41698	41703	41728	41787	41807	41826	41869	41901	42217	42259	42324
		42398	42411	42427	42435	42439	42443	42465	42530	42558	42598	42626	42644
		42667	42763	42912	43053	43177	43185	43198	43203	43208	46822	46838	
WBUS1T00		2800 #	6749	6868	7722	21972	24693	27112	27889	32586	33510	34355	34688
		37228	38293										
WBUS1T00.NE.0		2801 #	6586	20111	20117	23074	23115	23248	23265	23669	23884	23969	24809
		24820	25113	34991	35000	35009	35018	35098	36328	38088	38210		
WBUS31T030		2804 #	6608	7180	7203	7223	7615	7627	7635	7676	7796	7815	7908
		8122	8178	9405	9410	9435	9787	9809	9813	11657	11799	12332	12342
		12353	13586	13623	13652	13683	13707	14678	14992	15352	15793	16204	16263
		16487	16492	17621	20481	20604	20725	21282	22537	22920	22939	22976	23085
		23225	23253	23357	23479	23569	23573	23725	23829	23898	23957	24027	24061
		24115	24365	24520	24655	24802	24925	24959	25046	25091	25357	27097	27136
		27178	27183	27187	27257	27264	27275	27827	27834	27844	27885	28563	29116
		29463	29555	29877	29931	29938	30226	30997	31001	31007	31019	31025	32682
		33903	33993	34004	34842	34846	34850	34900	34904	34968	35043	35047	35059
		35071	35087	35140	35148	35152	35160	35164	35173	35186	35191	35196	35200
		35209	35218	35231	35235	35244	35261	35280	35304	35423	35465	35474	35539
		35558	35636	35640	35764	35787	35795	35806	35830	35846	35869	35877	35881
		35889	35901	35944	36038	36255	36631	37336	37419	37658	37705	38102	38400
		38467	39354	40055	40141	40399	40532	40538	41378	42683			
WBUS5T00		2802 #	6483	6773	6896	20225	27145	28038	28160	29124	29415	29575	32049
		32135	32550	33060	33065	33563	33643	33664	33777	33810	34720	34983	35252
		35320	35348	35759	35977	37666	37868	38367	38408	38453	38476	38533	38767
		38825	38928	39016	39049	39147	39158	41561	46402	46427	46452	46487	
WBUSNA		2817 #	39216	39264	39294								
WX.EQ.0		2806 #	5335	5500	5540	5574	5636	5682	5694	5742	5785	5807	5947
		6007	6229	6281	6285	6299	6313	6327	6341	6355	6369	6383	6397
		6411	6425	6499	6508	7055	7059	7071	7083	7166	7231	7318	7326
		7334	7342	7349	7357	7365	7373	7378	7482	7695	7699	7714	7718
		7746	7850	7924	7974	8068	8076	8080	8199	8920	9445	9698	9703
		9706	9711	9717	11145	11157	11466	12637	14734	15162	15170	15504	15510
		15698	15997	16461	16467	18236	19087	19154	19178	19237	19469	19478	19483
		20177	20350	20355	20360	20365	20370	20375	20379	20384	20389	20394	20399
		20404	20409	20540	20573	20608	20618	20627	20637	20647	20657	20667	20676
		20686	20696	20706	20716	20955	21385	21395	21604	21665	21698	21787	21814
		21868	21889	21952	21979	21994	22012	22021	22468	22478	22633	22684	22688
		22730	23056	23111	23261	23531	23536	23702	23767	23812	23880	23985	24048
		24074	24136	24216	24220	24227	24233	24308	24708	24797	24835	25612	26111
		26465	27788	27792	28086	28090	28477	28500	28554	28598	28797	29567	29660
		29664	29670	29674	29822	30249	30499	30648	30691	30696	30700	30735	30746
		30836	30848	30904	30942	30948	30983	30991	31013	31033	31039	31043	31062
		31112	32880	33027	33467	33673	33677	34021	34147	34536	34838	34864	34887
		34891	35095	35169	35205	35214	35240	35456	35532	35536	36031	36059	36479
		36496	36563	36580	36723	36749	37694	37698	37737	37744	39112	39347	39376
		41342	41346	41435	41439	41969	41973	42114	42695	46239	46247	46301	46309
		46372	46377	46382	46398	46423	46448	46472	46483	46529	46534	46539	46566

	46573	46580	46587	46593	46598	46605	46612	46619	46626	46632	46639	46646
	46652	46659	46665	46672	46679	46686	46693	46735	46883	46890	46919	46952
	47009	47019	47066	47091	47098	47109	47162	47173	47208	47219	47248	47259
	47299	47353	47375									
WX.NE.0	2807 #	5468	5716	5761	5824	6464	6471	6478	6647	6861	7314	7322
	7330	7338	7345	7353	7361	7369	7644	7648	7963	8031	8937	9796
	9800	9817	9821	9826	9844	11770	11781	11813	12526	12533	12561	12586
	14701	15072	15088	15923	16516	16991	17238	17482	17530	19083	19174	19242
	20598	20751	21716	21772	21907	21959	22473	23205	23333	23821	24004	24019
	24092	24107	24258	24264	24324	24635	24684	24764	25065	25801	25817	25827
	25837	25885	25889	25896	25907	25920	26303	27087	27118	27122	27126	27130
	27196	27332	27427	27754	27797	28594	28606	28618	28623	28633	28636	28640
	28765	28774	28793	29093	29142	29479	29542	29712	29717	29855	29949	30190
	30781	31456	31588	31650	32629	32635	32639	32645	32655	32671	32687	32702
	32707	32857	32862	32884	32916	32920	32938	32942	32948	32979	32992	32995
	33046	33050	33078	33082	34072	34089	34150	34260	35567	36052	36905	37719
	37831	37845	38152	39275	41355	42083	42479	46183	47574	47721	47732	47758
	48028	48037										
	2845 #											
CCOP1.CCBR_SIGND	2848 #	8260	8266	8278	8291	8306	8373	8377	8404	8436	8441	8471
	8535	8552	8563	8589	8595	8602	8619	8708	8726	8736	8915	8930
	8941	8986	9044	9049	9318	9328	9336	9339	9501	9506	9686	9721
	9726	9731	9736	9742	9748	9755	9773	10752	10822	10831	11016	11170
	11181	11261	11481	11488	11670	11677	11689	11698	11705	11775	11788	11898
	11909	11915	13166	13245	13773	13799	14316	14342	14722	14750	14816	14821
	15694	15730	15893	15910	16496	16539	16742	16755	16770	16811	16826	16950
	16956	17101	18235	18249	18321	19298	19342	19589	19594	21476	21554	22467
	22477	22536	22683	22687	22716	23702	23767	23811	23979	23984	23989	24040
	24047	24073	24083	24129	24135	24323	24376	24561	29748	29930	29937	40990
	40995	41000	41006	41026	41031	41037	41044	41064	41069	41075	41082	41139
	41144	41149	41155	41214	41232	41322	41328	41333	41413	41419	41425	41486
	41490	41494	41505	41567	41572	41657	41727	41776	41786	41806	41820	41825
	41830	41836	41853	41859	41868	42015	42028	42033	42148	42210	42216	42221
	42242	42248	42258	42275	42281	42291	42308	42314	42323	42538	42542	42546
	42566	42571	42620	42625	42630							
CCOP2.CCBR_SIGND	2849 #	8239	8244	8300	8315	8321	8351	8357	8362	8367	8416	8420
	8459	8497	8510	8522	8636	8652	8678	8684	8691	8763	8769	8776
	8807	8813	8820	8851	8857	8865	8946	8992	8995	9058	9066	9764
	9781	9835	9840	10694	10699	10704	10764	10774	11011	11020	11130	11136
	11140	16851	17478	17535	18279	18290	19292	19336	19390	19395	19450	19456
	21036	21041	21558	22462	22631	24189	24215	24219	24239	35629	35655	36517
	36522	41102	41107	41113	41120	41175	41180	41185	41192	41259	41264	41270
	41281	41525	41529	41533	41544	41600	41608	41613	41627	41885	41891	41900
	41916	41922	41931	42020	42038							
NOP.CCBR_ALUS	2847 #	8989	9104	9301	9310	9388	9454	9465	9471	9600	9634	14489
	15515	18460	18465	18470	18664	19529	25655	25843	25848	25851	25931	25936
	26353	26357	26603	26641	26767	26819	27007	27055	27254	28134	28511	28788
	29927	30126	30131	30137	30141	30370	30400	31522	31554	31557	31604	31635
	31719	42119	42125	42134								
NOP.CCBR_SIGND	2846 #	7032	8927	9027	9314	9441	9694	9829	11336	11352	12222	12556
	13467	15329	15334	15888	16996	17245	19303	19310	19315	19349	19353	19358
	19399	19405	21031	21046	21050	21143	23469	23694	24314	24343	24485	26172
	28248	28279	28323	28583	28589	29689	29731	30575	30580	30830	31494	33649
	34351	35272	35363	38833	40483	40488	40493	40622	41958	41963	41985	42098
	42105	46908	46990	47000	47337	47370	47751					

Cross Reference Listing - Field Names and Defined Values

CCMISC		2855 #										
	ALUS_DSDZ.CCBR_ALUS	2859 #	26348	28567	29919	30222	30897	30937	30947	30983	30991	31033	32536
		32555											
	ALUS_SIGND.CCBR_ALUS	2860 #	8898	8904	8981	9023	9099	9270	9275	9280	9286	9291	9296
		9381	15503	15509	18401	21026	31534	31539	31548	31574	41948	42082	45057
		47867											
	ALUS_UNSGN.CCBR_ALUS	2861 #	9553	9564	9580	9613	9618	9647	9652	14461	14479	15526	15531
		15536	15542	19519	19558	23144	23191	23296	23320	25921	25927	26764	26815
		27196	27207	27234	28489	31674	31692						
	NOP.CCBR_BRATST	2856 #	14077	14108	14471	14475	16629	24036	24125	24337	24457	26104	26444
		27075	27292	27304	27441	27450	27938	27964	27995	28030	28166	28746	29258
		29530	29624	30725	31129	31436	31460	31599	31630	32446	32467	32502	32525
		32733	32794	33572	34373	34388	36312						
	NOP.CCBR_CSIGNS	2857 #	8910	9039	9324	9416	9425	41953	42108				
	SETV.CCBR_SIGND	2862 #	8950	11498	11711	23479	29986	32691	35639	38540	41351	41443	42129
		42137	47566										
	WB_ATCR.CCBR_SIGND	2858 #	5381	5492	6135	6145	6695	7132	7955	8020	8036	8041	8046
		8982	8989	9112	9121	9401	9486	12633	14717	14745	14859	14875	15386
		16742	16755	16771	16782	16812	16827	16852	16868	16877	16883	16894	16917
		16933	16939	16950	16957	16980	17102	17254	17299	20156	20801	20950	21315
		21350	21412	21984	22025	22050	22099	23744	23748	23752	23792	23797	23858
		23862	23866	24661	25173	25725	26871	26880	28959	29209	29765	29846	30029
		31169	31309	31740	31748	31865	33245	33559	34704	35051	35055	35387	35543
		35651	35660	35779	35783	35802	35810	35897	35948	36010	36083	36088	36097
		36121	36248	36345	36350	36619	36796	36896	37182	37189	37208	37212	37213
		37218	37219	37336	37690	37709	37804	37827	37841	37864	37918	38035	38084
		38098	38149	38206	38219	38362	38449	38528	38529	38534	38556	38556	38763
		38817	38894	38898	38906	38924	38962	39034	39154	39944	39955	40858	41362
		41385	41561	41643	41664	41693	41698	41703	42041	42171	42508	42670	42683
		47278	47570										
CCPSL		2868 #											
	CC_WB.CCBR_ALUS	2870 #	10925	10929	10933	11029	11034	11149	11154	15417	15421	15425	16882
		17474	22054	25634	25796	25973	25994	26002	26006	26009	26017	26029	26142
		26146	26361	26364	26370	26373	26500	26504	27665	27686	29768	30033	30242
		30256	31312	31441	31498	31775	33533	33538	33720	33726	33737	34036	34185
		34285	34297	42086	42511	42515	47469	47473	47477	47539	47544	47549	
	MDR_USR.CCBR_BRATST	2873 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
		9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
		9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
		14235	14255	14673	15783	18866	18871	18876	18881	18886	18891	18896	18901
		18926	18930	18934	18938	18942	18946	18950	18954	18969	18974	18979	18984
		18989	18994	18999	19003	19606	19610	19614	19618	19622	19626	19630	19634
		19657	19662	19667	19672	19677	19682	19687	19691	38615	38621	39205	39244
		39263	39293	42335	42346	42426	42478	42503	42643	44362	45368	45455	45602
		45914	46914										
	PSL_WB.CCBR_ALUS	2871 #	5892	6221	6822	6886	6951	20486	34569	36482	36499	36503	36508
		36566	36583	36587	36799	36916	37340	37396	37413	37422	38267	38289	39427
		39430	40877	40881	40885	41366	42048	42052	42056	42060	42175	42673	
	PSW_WB.CCBR_ALUS	2872 #	20756	21389	21399	42661							
	WB_PSL.CCBR_SIGND	2869 #	6694	12633	15385	20155	20950	21350	21411	30029	33559	34704	35650
		35659	35778	36248	36345	36350	36618	36796	36895	37181	37187	37334	40857
		41362	41384	41560	41641	41662	41691	41696	41701	42041	42171	42507	42670
		42682	47570										
CLKX		2880 #											
	NOP	2881 #											

Cross Reference Listing - Field Names and Defined Values

	XTND	2882 #	20154	20552	21105	21209	21218	21593	21674	21689	21797	21884	29207
		29619	34883	34952	35632	35746	36268	36470	36488	36554	36572	36674	36772
		36848	36892	37236	37241	37266	37288	37409	38259	38418	38931	39172	39364
		39381	39437	39796	40028	40042	40075	40080	40158	40163	40198	40371	40383
		40421	40426	40554	40643	40687	40751	40810	40852				
DQ1		2884 #											
	D_WX	2887 #	5526	5614	5625	5630	5646	5712	5903	5947	6006	6170	6269
		6746	7853	8177	9309	9376	9481	9489	9573	9584	12221	12331	12341
		12371	13670	14450	14705	15886	18198	18217	18274	18286	18315	18330	18651
		19283	19329	19381	19448	19455	19571	19578	20110	20116	20127	20138	21020
		21588	21768	21864	22418	22596	22661	22710	22949	22994	23053	23110	23190
		23214	23234	23261	23319	23342	23483	23487	23530	23535	23693	23821	23880
		23907	23943	23992	23997	24001	24004	24019	24092	24107	24189	24215	24219
		24238	24323	24437	24778	24792	24843	25056	25084	25213	25312	25325	25608
		25686	25858	25863	26104	26180	26317	26321	26428	26451	26578	26588	26636
		27062	27067	27288	27336	27700	27747	27823	27964	27995	28027	28075	28154
		28394	29349	29859	29885	30495	30731	30771	30829	30987	30997	31001	31007
		31019	31025	31108	31415	31517	31714	32063	32067	32371	32379	32517	32682
		32790	32867	32870	33788	33792	33798	33813	33819	33831	34758	35331	35430
		35531	35535	35696	35896	36065	37098	38389	38404	38433	38443	38472	38698
		38705	38711	38716	38728	38756	39372	42107	42143	42153	42829	42975	46182
		46215	46301	46367	46386	46402	46427	46452	46487	46543	46729	46747	46780
		47069											
	NOP	2885 #											
	Q_D_WX	2888 #	5726	5749	22531	22541	24307	24515	37685	46212	46230	46246	46276
		46308											
	Q_WX	2886 #	5568	5573	5586	6260	6892	7725	8306	8332	8387	9038	9147
		9152	9157	9162	9305	9409	9440	9445	9558	9569	9791	9812	11129
		11139	11145	11157	13460	14048	14058	14069	14093	14102	14107	14660	14984
		14991	15077	15083	15093	15098	15126	15201	16029	16637	16654	17521	17629
		19444	21142	21281	21612	21712	22473	22630	22666	22715	23468	24546	24669
		24715	24831	25015	25098	25292	25296	25371	25655	26252	26280	26641	27356
		27360	27502	27512	27516	27525	27529	27533	27537	28038	28159	29415	30579
		30679	31522	36166	36687	36696	36755	36865	39215	40297	40324	41591	41619
		41801	41813	41863	41895	42102	42486	42739	42808	42820	42880	42966	43022
		43027	43108	43181	43213	44357	44361	44435	44442	44895	44902	44937	44944
		45016	45025	45274	45336	45364	45368	45455	45529	45539	45553	45817	45913
		45982	46076	46084	46238	46372	46377	46382	46410	46435	46460	46472	46528
		46533	46538	46563	47359	47381							
DQ2		2890 #											
	SQL	2891 #	11555	11644	14076	14097	15000	15836	15841	46482	46572	46579	46586
	SQL.D_WX	2893 #	46286										
	SQR	2892 #											
	SQR.D_WX	2894 #											
DQ3		2896 #											
	SQL.D_WX	2897 #	46226										
	SQR.D_WX	2898 #											
DTYPE		2900 #											
	BYTE	2901 #	6163	6591	6607	6651	6769	6778	6779	6782	6817	6900	6903
		6922	7107	7198	7219	7226	7317	7325	7333	7341	7386	7398	7401
		7413	7426	7429	7441	7450	7462	7472	7795	7814	7923	8088	9706
		9770	14650	14948	15917	16119	16212	16268	16511	16549	16995	17031	17053
		17243	17303	17499	18970	18975	18980	18985	18990	18995	19000	19004	19092
		19096	19097	19183	19187	19188	19248	19249	19253	19381	19444	19658	19663
		19668	19673	19678	19683	19688	19692	20734	20765	22513	22573	22614	22653

Cross Reference Listing - Field Names and Defined Values

	22751	23017	23021	23045	23090	23102	23258	23449	23453	23464	23468	23473
	23520	23730	23734	23756	23767	23786	23978	23988	24039	24082	24128	24195
	24201	24205	24226	24232	24257	24263	24313	24342	24457	24480	24692	24753
	24937	24983	25036	25172	25196	25262	25713	26302	26345	26895	27303	27520
	27674	27727	27752	27889	27937	28217	28296	28593	28618	28622	28632	28686
	28691	28703	28809	28824	28828	28831	28836	28849	28865	28945	28959	28962
	29093	29097	29103	29129	29145	29155	29163	29184	29207	29227	29234	29257
	29276	29284	29334	29339	29350	29369	29391	29421	29427	29619	29702	29706
	29841	29855	29910	30117	30125	30130	30205	30214	30221	30266	30272	30285
	30289	30346	30356	30359	30363	30373	30377	30387	30450	30484	30531	30535
	30579	30609	30618	30644	30668	30709	30746	30852	30896	30902	30924	30936
	30941	30982	30990	31032	31166	31169	31214	31248	31410	31431	31578	31673
	31744	31857	31864	31892	31920	31947	32037	32040	32112	32115	32119	32139
	32143	32152	32155	32159	32434	32442	32445	32489	32498	32925	32929	33011
	33131	33146	33155	33164	33172	33245	33499	33503	33513	33571	33623	33715
	33720	33726	33731	33737	33783	33952	34255	34297	34309	34368	34378	34435
	34446	34450	34494	34509	34613	34710	34714	34719	35256	35450	35469	35482
	35546	35579	35632	36019	36057	36065	36160	36299	36470	36488	36554	36572
	36631	36635	36639	38418	38614	38733	39002	39006	39015	39048	39054	39061
	39097	39104	39146	39796	40028	40042	40199	40371	40383	40555	40810	40853
	41776	41786	41801	41803	45817	46394	46419	46444	46908	46915	46989	46996
	47015	47018	47396									
IDEP	2904 #	8240	8245	8260	8265	8278	8292	8297	8301	8307	8312	8316
	8332	8351	8356	8363	8368	8373	8378	8387	8404	8436	8440	8497
	8501	8511	8522	8527	8553	8558	8564	8590	8596	8602	8606	8620
	8636	8644	8653	8679	8685	8691	8695	8709	8727	8731	8737	8763
	8769	8776	8780	8807	8813	8820	8824	8851	8857	8865	8869	8898
	8904	8909	8914	8926	8931	8942	8946	8982	8986	8992	8995	9023
	9043	9048	9059	9067	9099	9148	9153	9158	9163	9169	9174	9179
	9184	9190	9195	9199	9203	9208	9213	9218	9223	9229	9234	9238
	9242	9270	9275	9280	9286	9291	9296	9301	9305	9314	9319	9323
	9328	9335	9340	9382	9502	9506	9515	9519	9523	9527	9553	9564
	9580	9590	9595	9600	9607	9613	9618	9624	9629	9634	9641	9647
	9652	9687	9693	9720	9726	9731	9737	9743	9748	9755	9829	10753
	10765	10775	11130	11135	11140	11169	11180	11259	11320	11335	11351	11446
	11465	11471	11479	11486	11562	11668	11676	11687	11697	11704	11897	11908
	11914	13167	13246	13772	13797	14315	14340	14722	14750	15487	16741	16754
	16769	16810	16825	16850	16866	16875	16949	16955	18197	18216	18236	18240
	18244	18249	18273	18279	18287	18291	18314	18321	18329	18711	18721	18733
	18915	19291	19297	19303	19310	19315	19336	19342	19349	19353	19358	19389
	19395	19400	19404	19449	19455	19519	19525	19558	19571	19578	19589	19594
	21209	21218	21593	21674	21689	21797	21884	36517	36522	40687	40714	40719
	40990	41000	41006	41026	41037	41044	41064	41075	41082	41102	41113	41120
	41139	41144	41149	41155	41175	41185	41192	41206	41214	41225	41232	41251
	41259	41270	41276	41281	41305	41314	41322	41328	41333	41341	41405	41448
	41452	41486	41494	41499	41504	41525	41533	41538	41543	41567	41572	41591
	41600	41608	41613	41619	41627	41656	41727	41836	41853	41859	41868	41885
	41891	41900	41916	41922	41931	41948	41953	41958	41963	41984	42015	42020
	42028	42033	42038	42082	42098	42102	42108	42148	42158	42163	42210	42216
	42221	42242	42248	42258	42275	42281	42291	42308	42314	42323	42357	42375
	42383	42388	42393	42438	42447	42478	42772	42819	42830	42920	42965	42976
	43061	43116	43214	43242	43251	44357	44367	44373	44378	44392	44397	44415
	44424	44436	44476	44483	44489	44493	44505	44508	44523	44527	44534	44550
	44571	44578	44584	44589	44601	44604	44619	44623	44661	44668	44674	44679
	44691	44694	44709	44714	44717	44757	44763	44767	44779	44782	44798	44805

Cross Reference Listing - Field Names and Defined Values

	44835	44841	44846	44860	44865	44883	44888	44889	44896	44950	44955	44969
	44973	45001	45006	45010	45031	45073	45079	45084	45097	45102	45142	45147
	45185	45193	45200	45216	45221	45234	45290	45296	45301	45314	45319	45364
	45373	45379	45384	45398	45403	45419	45451	45460	45466	45471	45485	45490
	45508	45523	45529	45538	45598	45607	45612	45617	45631	45636	45652	45682
	45717	45722	45727	45741	45746	45764	45772	45832	45838	45843	45857	45862
	45880	45908	45919	45926	45932	45948	45954	45972	46023	46024	46029	46034
	46048	46053	46069	46075	47082							
LONG	2903 #	5545	5546	5552	5563	5568	5578	5595	5621	5630	5671	5686
	5730	5753	5789	5799	5817	6498	6512	6516	6580	6600	6620	6794
	6795	6814	6843	6914	6917	7510	7799	7818	8416	8420	8455	8459
	8467	8471	8536	9027	9038	9072	9409	9415	9424	9440	9764	9774
	9782	9835	9840	10693	10698	10703	10821	10830	11011	11015	11020	11223
	11657	11776	11789	11803	11929	11934	12850	13080	13674	13678	14208	14219
	14234	14254	14461	14479	14671	14768	14773	14778	14783	14803	14808	14815
	14820	14953	15319	15329	15334	15494	15503	15509	15526	15531	15536	15542
	15694	15718	15730	15758	15782	15800	15851	15893	15910	16019	16030	16033
	16074	16217	16497	16533	16539	16568	16593	16609	16616	16638	16642	16782
	16893	16932	16938	17009	17014	17019	17040	17081	17089	17100	17260	17265
	17284	17477	17492	17535	17556	18401	19009	19075	19109	19113	19117	19121
	19125	19129	19639	19644	19698	19722	19727	19738	20154	20190	20195	20197
	20229	20239	20248	20257	20265	20274	20283	20292	20300	20309	20318	20327
	20335	20344	20418	20424	20433	20441	20444	20454	20468	20475	20503	20508
	20513	20530	20535	20552	20569	20580	20585	20590	20611	20620	20630	20640
	20650	20660	20669	20679	20689	20699	20709	20718	20760	20768	20815	20820
	21026	21031	21036	21041	21046	21050	21147	21147	21152	21158	21272	21362
	21446	21449	21457	21464	21476	21482	21527	21530	21531	21538	21539	21550
	21554	21558	21611	21618	21624	21659	21705	21711	21726	21734	21758	21779
	21805	21812	21848	21861	21877	21900	21907	21920	21965	21999	22003	22015
	22049	22057	22092	22428	22462	22467	22477	22545	22632	22761	22781	22917
	22973	23085	23144	23154	23158	23182	23190	23196	23201	23253	23295	23306
	23310	23319	23324	23329	23680	23693	23703	23725	23782	23803	23812	23821
	23825	23829	23833	23838	23844	23848	23932	23984	24019	24027	24047	24073
	24107	24115	24135	24190	24240	24485	24560	24644	25001	25095	25288	25649
	25791	25796	25884	25890	25895	25900	25906	25912	25919	25926	26172	26212
	26582	26607	26699	26704	26709	26714	26730	26735	26740	26745	26757	26760
	26763	26808	26811	26814	26827	26835	26838	26841	27160	27164	27168	27172
	27195	27206	27216	27226	27229	27233	27431	27435	27446	27449	27494	27502
	27768	27945	27995	28014	28060	28078	28151	28171	28249	28278	28323	28328
	28401	28411	28418	28457	28461	28469	28484	28488	28522	28524	28527	28584
	28588	28846	28942	28978	28982	29120	29180	29223	29253	29280	29397	29424
	29511	29514	29519	29529	29689	29692	29731	29734	29748	29869	29902	29930
	29937	30000	30045	30243	30257	30575	30585	30625	30830	30864	30973	30978
	31072	31175	31221	31495	31533	31538	31678	31682	31685	31691	31860	31871
	31914	31932	32034	32081	32123	32407	32411	32501	32692	32843	32847	32888
	32893	32906	33001	33006	33030	33041	33119	33167	33610	33615	33638	33649
	33835	34036	34185	34285	34304	34350	34462	34501	34663	34736	34740	34744
	35271	35629	35655	35951	36034	36255	36341	36357	36360	36684	36776	36812
	36817	36870	36878	36881	36891	37330	37353	37356	37360	37364	37368	37479
	37678	37980	37989	37993	37997	38001	38005	38009	38039	38366	38452	38482
	38486	38566	38571	38658	38662	38775	39074	39166	39184	39334	39335	39338
	39341	40483	40488	40493	40622	40751	40774	40778	41413	41419	41424	42652
	44382	44443	44454	44496	44538	44544	44592	44634	44640	44682	44723	44729
	44770	44850	44903	44913	44959	45026	45038	45057	45088	45131	45136	45205
	45239	45245	45305	45337	45388	45475	45548	45552	45564	45621	45663	45731

		45780	45789	45800	45847	45936	45987	46038	46083	46170	46175	46367	46729
		46748	46767	46870	46878	46882	46889	47008	47052	47059	47150	47196	47236
		47331	47337	47345	47360	47365	47370	47382	47866				
WORD		2902 #	5467	6138	6166	6436	6617	6656	6701	6786	6787	6790	6907
		6910	7031	7348	7356	7364	7444	7485	7488	7495	7507	7517	7529
		7539	7967	8050	8147	8194	8320	11114	11166	11177	11386	11458	11808
		11812	11816	12222	12237	12555	12560	12606	12620	12627	12844	12858	13062
		13085	13312	13323	13355	13366	13467	13871	13876	13949	13959	13968	13977
		14430	14442	14645	14689	14711	14734	14826	14977	15007	15014	15078	15094
		15161	15351	15362	15366	15371	15375	15406	15476	15519	15817	15823	15828
		15875	15880	15888	15986	16014	16287	16474	16487	17071	17455	17578	17601
		17626	17638	19283	19329	19538	19544	19565	20109	20115	20145	20159	20208
		20220	20234	20480	20738	21107	21127	21132	21142	21244	21249	21263	22417
		22497	22524	22537	22556	22596	22660	22683	22687	22698	22709	22716	22764
		24215	24219	24323	24376	24447	24640	24675	24721	24747	24769	24785	24793
		24815	24825	24877	24949	24968	24973	24998	25030	25061	25070	25085	25099
		25103	25161	25210	25214	25276	25313	25319	25326	25382	25402	25600	26089
		26100	26419	26425	26441	27275	27298	27506	27711	27737	27820	27941	27964
		27970	27979	27989	27992	28003	28017	28021	28240	28971	29100	29109	29346
		29555	29566	29570	29574	29640	29809	30083	30099	30174	30189	30324	30336
		30427	30438	30664	30673	30723	30856	30860	30946	30958	30963	30967	31178
		31238	31423	31436	31548	31574	31881	32085	32455	32463	32466	32474	32485
		32963	32964	32968	32969	33134	33175	33466	33474	33993	34008	34012	34607
		34620	34623	34654	34670	34676	35026	35275	35308	35319	35334	35347	35360
		35363	35440	35445	35971	35976	35983	36015	36072	36146	36296	36465	36549
		37143	37344	37491	38168	38539	38620	38822	38827	38833	38837	38845	39024
		39397	39507	40995	41031	41069	41107	41180	41264	41489	41528	41604	41762
		41771	41782	41821	41825	41831	42402	42537	42542	42545	42550	42557	42565
		42570	42579	42603	42619	42625	42629	42655	42658	42715	42720	42767	42776
		42782	42792	42797	42839	42845	42859	42865	42916	42925	42931	42942	42947
		42985	42991	43006	43057	43066	43072	43083	43087	43126	43130	43152	43158
		43166	43197	43202	43207	43226	43231	43236	45677	46999	47629	47639	47679
		47689	47750	47781	47797	47814	47886	47889	47892	48027			
FPA		2906 #											
	FPA_DATA.MBUS	2909 #	8972	8976	16727	16746	16786	16816	16838	16858	16905	16968	17018
		17080	17085	17095	17195	17199	17203	17210	17214	17218	17221	17224	17269
		44348	44352	44386	44401	44409	44430	44435	44442	44819	44824	44855	44869
		44877	44895	44902	45354	45359	45368	45392	45407	45415	45539	45547	45553
		45780	45898	45903	45959	45967	45977	45982	45990	46008	46012	46057	46065
		46076	46084										
	FPA_DATA.WBUS	2910 #	16791	16802									
	FPA_MBUS.FPA_WBUS	2912 #	16722	16733	16761	16797	16833	16844					
	FPA_MBUS.LITNXT	2911 #	44361	45913									
	NOP	2907 #											
	WBUS_FPA	2913 #	8981	8989	9111	9121	16740	16753	16767	16780	16808	16823	16851
		16865	16874	16892	16931	16937	16948	16954	17100	17253	17299		
	WBUS_FPA.CC	2914 #	16882										
ISTRM		2921 #											
	ISIZE_DSIZE	2923 #	5467	5671	7031	7198	9195	15406	17499	18970	18975	18980	18985
		18990	18995	19000	19004	19092	19096	19183	19187	19248	19253	19283	19329
		19381	19444	19658	19663	19668	19673	19678	19683	19688	19692	20480	23017
		23021	23090	23154	23158	23449	23453	23464	23468	23730	23734	23782	23786
		23844	23848	23978	23984	23988	24039	24047	24073	24082	24128	24135	24195
		24313	24342	26302	27752	27979	28003	28060	28686	28691	30205	30484	30579
		30644	30664	30668	30673	30723	30896	30902	30936	30941	30946	30982	30990

		31032	33146	33571	33623	42767	42792	42797	42859	42916	42942	42947	43057
		43083	43087	43152	43197	43202	43207	44357	44382	44476	44496	44571	44592
		44661	44682	44770	44850	44959	45006	45088	45142	45205	45305	45364	45388
		45451	45475	45529	45598	45621	45682	45731	45817	45847	45908	45936	45987
		46038	46882	46889	46989	46999	47008	47015					
	NOP	2922 #											
JSR		2925 #											
	NOP	2926 #											
	PUSH	2927 #	5343	5349	5354	5371	5376	5418	5427	5432	5442	5477	5564
		5569	5579	5590	5596	5656	5896	5910	5917	5923	5932	5990	6149
		6156	6160	6167	6171	6225	6256	6261	6270	6274	6289	6303	6317
		6331	6345	6359	6373	6387	6401	6415	6456	6491	6513	6520	6523
		6542	6548	6552	6581	6596	6690	6698	6757	6840	6960	6974	7126
		7135	7162	7177	7199	7208	7220	7264	7268	7272	7276	7280	7284
		7291	7306	7393	7420	7445	7466	7489	7511	7533	7553	7561	7565
		7585	7607	7611	7734	7750	7765	7846	7861	7873	7920	8008	8036
		8094	8120	8174	8187	8191	8195	8496	8601	8690	8775	8819	8863
		8903	8909	8976	9018	9022	9038	9095	9098	9305	9323	9376	9381
		9415	9424	9494	9665	10747	10977	11320	11335	11441	11455	11552	11556
		11630	11641	11646	11794	11903	11978	11984	12223	12338	12380	12406	12550
		12591	12599	12616	12834	12840	12846	12865	13053	13058	13091	13153	13157
		13229	13233	13241	13450	13454	13462	13468	13572	13577	13587	13611	13624
		13763	13768	13782	13786	13791	13863	13872	13877	14049	14059	14070	14088
		14118	14213	14229	14248	14306	14311	14325	14329	14334	14420	14430	14451
		14466	14484	14640	14646	14652	14657	14662	14666	14702	14713	14717	14740
		14745	14757	14852	14937	14949	14954	14961	14971	14979	14986	15061	15192
		15202	15300	15305	15310	15315	15386	15478	15482	15489	15665	15690	15693
		15711	15726	15742	15755	15771	15777	15789	15889	15900	15905	15986	15992
		16199	16235	16238	16248	16256	16274	16420	16441	16448	16453	16483	16488
		16493	16502	16516	16525	16534	16562	16656	16734	16747	16762	16803	16817
		16845	16877	16902	16907	16912	16923	16965	16969	16975	16987	16997	17010
		17026	17041	17044	17072	17106	17196	17200	17211	17215	17233	17245	17261
		17274	17286	17289	17449	17457	17462	17504	17531	17551	17609	18197	18216
		18273	18286	18314	18329	18367	18395	18651	18655	18675	19048	19061	19071
		19279	19514	20161	20230	20240	20249	20258	20266	20275	20284	20293	20301
		20310	20319	20328	20336	20345	20425	20434	20450	20455	20557	20614	20623
		20633	20643	20653	20663	20672	20682	20692	20702	20712	20721	21007	21012
		21017	21022	21032	21108	21112	21123	21144	21210	21222	21240	21386	21396
		21439	21443	21450	21468	21472	21518	21532	21543	21585	21594	21651	21675
		21690	21764	21798	21854	21885	22098	22424	22439	22445	22508	22512	22523
		22530	22566	22572	22579	22602	22608	22621	22659	22672	22704	22708	22715
		22720	22928	23025	23162	23314	23457	23492	23679	23738	23852	23921	23931
		24051	24139	24209	24346	24366	24442	24461	24466	24498	24524	24530	24751
		24773	24842	24858	24943	24967	25029	25034	25074	25160	25202	25248	25282
		25287	25607	25660	25664	25691	25802	25806	25810	26096	26100	26114	26419
		26424	26435	26440	26455	26593	26597	26640	26645	27022	27043	27048	27090
		27101	27140	27319	27332	27341	27458	27469	27499	27503	27513	27517	27526
		27530	27534	27704	27709	27764	27959	28013	28043	28105	28116	28177	28211
		28296	28402	28406	28431	28465	28713	28717	28729	28733	28737	28754	28941
		28979	29138	29154	29181	29213	29235	29250	29254	29269	29281	29339	29355
		29459	29504	29546	29550	29620	29632	29921	30091	30096	30101	30105	30109
		30182	30187	30197	30202	30206	30333	30338	30352	30435	30440	30481	30571
		30670	30685	30823	30857	30861	30869	30952	31138	31410	31423	31426	31431
		31447	31473	31527	31549	31607	31638	31719	32196	32211	32233	32248	32270
		32284	32415	32540	32624	32741	32791	32818	33484	33489	33567	33899	34017

		34068	34085	34141	34146	34249	34254	34287	34398	34455	34600	34953	35321
		35349	35516	35540	35576	35585	35590	35747	35919	35923	35927	35931	35935
		35939	35978	36039	36300	36304	36320	36354	36466	36475	36479	36492	36496
		36550	36559	36563	36576	36580	36627	36675	36679	36692	36770	36809	36849
		37099	37357	37365	37488	37560	37751	37764	37769	37788	37864	37873	37878
		37962	37965	37969	37973	37977	37986	37994	37998	38002	38006	38035	38428
		38483	38563	38579	38655	38763	38772	38817	38898	38924	39154	39163	39181
		39386	39469	39478	39674	39684	39726	39740	39802	39885	39920	39976	40056
		40142	40267	40400	40408	40693	40715	40756	40775	40815	40874	40902	40930
		41954	42108	44410	44477	44572	44662	44878	44986	45115	45416	45502	45648
		45758	45875	45968	46065	46145	46152	46156	46176	46815	46832	46871	46879
		46986	47053	47087	47105	47158	47169	47201	47215	47241	47255	47332	47361
		47398	47402	47412	47423	47426	47725	47737	47745	47777	47960	48049	
LIT		2929 #											
	FPAWAIT	2932 #	8981	8989	9110	9120	16739	16752	16766	16779	16807	16822	16850
		16864	16873	16881	16891	16930	16936	16947	16953	17090	17096	17099	17285
	LITRL	2931 #	5327	5333	5357	5365	5384	5398	5402	5408	5454	5458	5462
		5471	5476	5488	5495	5499	5503	5507	5515	5519	5523	5530	5534
		5539	5559	5582	5590	5600	5605	5614	5617	5625	5630	5659	5662
		5675	5681	5687	5698	5712	5715	5723	5731	5749	5784	5796	5823
		5898	5912	5917	5933	5939	5943	5951	5954	5957	5963	5966	5977
		5991	5997	6006	6127	6132	6142	6148	6155	6159	6170	6174	6221
		6224	6228	6232	6235	6239	6243	6247	6255	6264	6269	6273	6280
		6284	6288	6292	6298	6302	6306	6312	6316	6320	6326	6330	6334
		6340	6344	6348	6354	6358	6362	6368	6372	6376	6382	6386	6390
		6396	6400	6404	6410	6414	6418	6424	6432	6463	6470	6477	6486
		6502	6507	6527	6535	6541	6544	6556	6612	6660	6665	6707	6752
		6803	6825	6831	6857	6931	6964	6977	7013	7036	7041	7048	7051
		7054	7058	7062	7065	7070	7074	7077	7082	7086	7089	7094	7098
		7101	7106	7110	7165	7169	7173	7176	7183	7187	7230	7234	7263
		7267	7275	7279	7283	7309	7313	7321	7329	7337	7344	7352	7360
		7368	7372	7376	7382	7404	7409	7419	7432	7437	7453	7458	7465
		7475	7480	7498	7503	7520	7525	7532	7542	7547	7556	7560	7564
		7568	7577	7581	7591	7594	7614	7626	7634	7643	7647	7651	7663
		7675	7689	7705	7713	7717	7729	7740	7745	7749	7753	7756	7761
		7768	7789	7809	7828	7832	7849	7860	7871	7907	7911	7915	7943
		7949	7958	7962	7973	7985	8008	8013	8024	8030	8057	8062	8067
		8072	8075	8079	8093	8143	8186	8190	8198	8205	9395	9399	9420
		9430	9449	9459	9558	9569	9669	9690	9770	9778	10816	10825	10867
		10878	10925	10929	10933	11029	11034	11149	11154	11326	11340	11768	11780
		11798	11811	11885	12033	12040	12236	12331	12341	12371	12384	12399	12510
		12516	12549	12559	12579	12594	12619	12636	12641	12699	12703	12709	12715
		12721	12724	12729	12734	12857	12876	13084	13478	13585	13622	13670	13682
		13702	13875	14086	14112	14116	14424	14441	14446	14500	14505	14665	14687
		14700	14710	14733	14835	14840	14970	15005	15012	15071	15077	15083	15087
		15093	15098	15123	15166	15389	15417	15421	15425	15689	15710	15740	15775
		15811	15827	15846	15874	15879	15903	15996	16059	16063	16069	16113	16118
		16136	16140	16198	16230	16290	16435	16482	16524	16560	16574	16598	16603
		16622	16666	16990	17117	17122	17159	17168	17172	17176	17237	17474	17562
		17565	17592	17608	17634	19041	19052	19064	19076	19153	19166	19236	19241
		19468	19477	19482	19582	20123	20127	20134	20138	20156	20175	20200	20211
		20243	20252	20261	20269	20278	20287	20296	20304	20313	20322	20331	20339
		20349	20354	20359	20364	20369	20374	20378	20383	20388	20393	20398	20403
		20408	20428	20437	20449	20453	20465	20468	20490	20494	20498	20536	20539
		20544	20548	20553	20572	20607	20613	20617	20622	20626	20632	20636	20642

20646	20652	20656	20662	20666	20671	20675	20681	20685	20691	20695	20701
20705	20711	20715	20720	20742	20750	20778	20786	20791	20804	20810	20899
20913	20929	20954	20962	20965	20972	20975	21102	21115	21136	21164	21206
21213	21262	21318	21324	21327	21384	21394	21438	21453	21467	21471	21542
21546	21603	21615	21621	21655	21664	21697	21730	21758	21786	21794	21808
21848	21858	21867	21874	21888	21903	21910	21951	21968	21978	21997	22002
22011	22020	22428	22481	22485	22545	22637	22677	22785	22999	23004	23030
23034	23038	23049	23078	23095	23106	23128	23167	23171	23175	23186	23204
23237	23245	23278	23332	23437	23688	23760	23807	23816	23922	23946	23957
24014	24061	24102	24250	24365	24422	24462	24471	24477	24494	24525	24540
24647	24664	24669	24680	24683	24707	24760	24782	24796	24801	24805	24834
24868	24929	24933	24952	24959	25025	25040	25045	25050	25090	25109	25119
25165	25203	25243	25283	25292	25299	25367	25611	25634	25648	25676	25680
25713	25790	25796	25973	25994	26002	26006	26009	26017	26029	26110	26142
26146	26211	26293	26361	26364	26370	26373	26464	26500	26504	26581	26789
26793	26847	26852	26895	27013	27017	27022	27026	27043	27051	27062	27067
27101	27331	27374	27378	27665	27686	27731	27774	27778	27796	27800	27817
27827	27834	27838	27844	27848	27854	27858	27868	27872	27876	27911	27915
27983	28007	28027	28048	28064	28075	28106	28117	28143	28147	28154	28176
28182	28187	28213	28245	28263	28268	28284	28296	28311	28327	28333	28337
28341	28417	28427	28453	28476	28493	28499	28506	28532	28537	28562	28597
28601	28617	28660	28665	28669	28769	28773	28778	28783	28792	28796	28823
28827	28835	28855	29074	29089	29112	29115	29133	29137	29141	29176	29246
29249	29268	29380	29488	29491	29511	29534	29538	29559	29565	29584	29589
29594	29599	29604	29636	29659	29669	29679	29684	29697	29710	29715	29721
29726	29818	29821	29828	29836	29842	29849	29868	29876	29911	30104	30201
30225	30242	30247	30252	30256	30265	30271	30276	30280	30284	30295	30351
30373	30377	30387	30451	30456	30460	30480	30491	30498	30518	30531	30584
30599	30609	30618	30624	30630	30635	30640	30713	30734	30755	30759	30780
30785	30873	30878	30887	30908	30926	30987	30997	31001	31007	31012	31019
31025	31048	31061	31111	31227	31230	31241	31348	31441	31455	31498	31504
31514	31587	31649	31688	31703	31707	31744	31926	32044	32080	32099	32122
32131	32162	32172	32371	32379	32384	32395	32399	32403	32420	32423	32426
32438	32459	32478	32482	32494	32511	32545	32564	32567	32570	32573	32590
32594	32598	32602	32607	32611	32615	32624	32628	32634	32638	32644	32648
32654	32658	32663	32667	32670	32686	32701	32706	32746	32790	32813	32822
32828	32832	32836	32861	32878	32883	32903	32915	32919	32937	32952	32983
32991	32996	33018	33026	33037	33051	33060	33064	33128	33150	33465	33482
33527	33533	33538	33541	33545	33575	33578	33609	33614	33618	33629	33637
33648	33653	33659	33668	33672	33676	33819	33831	33902	33916	33956	33998
34003	34028	34033	34071	34075	34088	34092	34132	34145	34149	34153	34157
34167	34171	34177	34197	34240	34259	34263	34267	34273	34294	34301	34318
34362	34379	34393	34405	34440	34458	34498	34504	34560	34569	34610	34658
34679	34682	34701	34707	34713	34837	34863	34890	34899	34903	34907	34912
35013	35030	35062	35066	35078	35082	35090	35106	35110	35114	35118	35122
35126	35176	35185	35204	35251	35268	35287	35314	35328	35331	35368	35372
35375	35381	35385	35390	35393	35402	35405	35408	35411	35417	35436	35455
35459	35468	35477	35481	35488	35492	35497	35502	35511	35515	35520	35525
35531	35535	35566	35570	35574	35584	35588	35598	35669	35696	35701	35741
35778	35782	35797	35801	35813	35817	35821	35825	35833	35837	35841	35849
35852	35856	35860	35872	35896	35905	35913	35918	35922	35926	35930	35934
35938	35986	35999	36004	36007	36046	36051	36060	36063	36082	36088	36092
36096	36101	36106	36111	36115	36120	36136	36139	36151	36156	36183	36227
36232	36237	36241	36248	36251	36287	36303	36315	36335	36353	36478	36495
36562	36579	36615	36619	36626	36667	36678	36683	36700	36715	36732	36745

Cross Reference Listing - Field Names and Defined Values

		36755	36781	36799	36808	36841	36847	36853	36871	36882	36904	36908	36916
		37098	37104	37109	37114	37119	37129	37135	37194	37204	37246	37251	37255
		37270	37274	37278	37282	37311	37316	37321	37374	37379	37384	37388	37400
		37426	37431	37436	37440	37486	37536	37559	37563	37585	37628	37634	37640
		37646	37661	37671	37685	37693	37697	37718	37722	37726	37736	37743	37768
		37787	37801	37807	37813	37830	37839	37844	37853	37856	37867	37876	37881
		37914	37958	37961	37964	37968	37972	37976	38009	38064	38084	38091	38095
		38105	38109	38112	38152	38156	38173	38202	38205	38213	38217	38222	38225
		38228	38270	38278	38282	38345	38351	38370	38379	38384	38389	38404	38419
		38427	38433	38443	38472	38546	38551	38561	38570	38577	38609	38626	38631
		38637	38643	38653	38698	38705	38711	38716	38721	38728	38739	38745	38751
		38756	38766	38806	38850	38855	38916	38927	39038	39120	39157	39209	39268
		39274	39375	39385	39430	39468	39484	39725	39739	39759	39805	39813	39833
		39923	39930	39948	39959	40086	40090	40149	40153	40168	40172	40179	40190
		40194	40217	40225	40232	40237	40241	40303	40309	40324	40327	40406	40431
		40435	40497	40502	40509	40515	40527	40541	40547	40551	40560	40571	40576
		40589	40593	40597	40605	40625	40646	40650	41030	41068	41210	41228	41310
		41336	41354	41366	41428	41642	41647	41651	41663	41668	41672	41678	41683
		41692	41697	41702	42044	42052	42056	42060	42086	42113	42175	42486	42661
		42673	42694	42788	42938	43079	43190	44750	45168	45603	45673	45785	46148
		46151	46155	46162	46165	46179	46182	46201	46218	46243	46251	46279	46305
		46313	46344	46352	46367	46386	46391	46402	46407	46416	46427	46432	46441
		46452	46457	46475	46478	46487	46492	46543	46569	46576	46583	46590	46595
		46601	46608	46615	46622	46629	46635	46642	46648	46655	46662	46668	46675
		46682	46689	46696	46716	46724	46729	46739	46744	46747	46763	46780	46807
		46810	46814	46818	46827	46831	46874	46886	46893	46895	46911	46922	46925
		46947	46951	46993	47003	47012	47022	47062	47069	47090	47094	47097	47101
		47108	47112	47135	47139	47153	47157	47161	47165	47168	47172	47176	47181
		47187	47190	47197	47200	47207	47211	47214	47218	47222	47227	47230	47237
		47240	47247	47251	47254	47258	47262	47265	47271	47281	47293	47298	47302
		47336	47356	47369	47378	47393	47401	47405	47411	47415	47422	47430	47469
		47473	47477	47493	47498	47503	47508	47515	47519	47539	47544	47549	47559
		47573	47646	47651	47654	47659	47740	47757	47780	47804	47808	47899	47903
		47935	47942	47947	47982	48032	48084	48087	48090	48095	48096	48099	
	LONLIT	2933 #	5381	5435	5445	5492	5608	5640	5664	5815	6135	6145	6695
		6836	7024	7132	7955	8020	8036	8041	8046	8982	8989	9112	9121
		9401	9486	12633	14717	14745	14859	14875	15386	16742	16755	16771	16782
		16812	16827	16852	16868	16877	16883	16894	16917	16933	16939	16950	16957
		16980	17102	17254	17299	20156	20801	20950	21315	21350	21355	21412	21984
		22025	22050	22099	23744	23748	23752	23792	23797	23858	23862	23866	24661
		25173	25725	26871	26880	28959	29080	29209	29765	29846	30029	30564	30570
		30822	31169	31309	31740	31748	31835	33245	33559	34704	34995	35004	35022
		35051	35055	35387	35543	35651	35660	35779	35783	35802	35810	35897	35948
		35964	36010	36083	36088	36097	36121	36248	36345	36350	36619	36712	36742
		36796	36896	37182	37189	37208	37213	37219	37336	37349	37690	37709	37804
		37827	37841	37864	37918	38035	38084	38098	38149	38206	38219	38362	38449
		38529	38534	38556	38763	38817	38894	38898	38906	38924	38962	39034	39154
		39344	39368	39944	39955	40858	41362	41385	41561	41643	41664	41693	41698
		41703	42041	42171	42508	42670	42683	46290	46977	47006	47049	47056	47278
		47323	47570										
	NORMAL	2930 #											
LITRL		2935 #	5327	5333	5357	5365	5384	5398	5402	5408	5454	5458	5462
		5471	5476	5488	5495	5499	5503	5507	5515	5519	5523	5530	5534
		5539	5559	5582	5590	5600	5605	5614	5617	5625	5630	5659	5662
		5675	5681	5687	5698	5712	5715	5723	5731	5749	5784	5796	5823

5898	5912	5917	5933	5939	5943	5951	5954	5957	5963	5966	5977
5991	5997	6006	6127	6132	6142	6148	6155	6159	6170	6174	6221
6224	6228	6232	6235	6239	6243	6247	6255	6264	6269	6273	6280
6284	6288	6292	6298	6302	6306	6312	6316	6320	6326	6330	6334
6340	6344	6348	6354	6358	6362	6368	6372	6376	6382	6386	6390
6396	6400	6404	6410	6414	6418	6424	6432	6463	6470	6477	6486
6502	6507	6527	6535	6541	6544	6556	6612	6660	6665	6707	6752
6803	6825	6831	6857	6931	6964	6977	7013	7036	7041	7048	7051
7054	7058	7062	7065	7070	7074	7077	7082	7086	7089	7094	7098
7101	7106	7110	7165	7169	7173	7176	7183	7187	7230	7234	7263
7267	7275	7279	7283	7309	7313	7321	7329	7337	7344	7352	7360
7368	7372	7376	7382	7404	7409	7419	7432	7437	7453	7458	7465
7475	7480	7498	7503	7520	7525	7532	7542	7547	7556	7560	7564
7568	7577	7581	7591	7594	7614	7626	7634	7643	7647	7651	7663
7675	7689	7705	7713	7717	7729	7740	7745	7749	7753	7756	7761
7768	7789	7809	7828	7832	7849	7860	7871	7907	7911	7915	7943
7949	7958	7962	7973	7985	8008	8013	8024	8030	8057	8062	8067
8072	8075	8079	8093	8143	8186	8190	8198	8205	9395	9399	9420
9430	9449	9459	9558	9569	9669	9690	9770	9778	10816	10825	10867
10878	10925	10929	10933	11029	11034	11149	11154	11326	11340	11768	11780
11798	11811	11885	12033	12040	12236	12331	12341	12371	12384	12399	12510
12516	12549	12559	12579	12594	12619	12636	12641	12699	12703	12709	12715
12721	12724	12729	12734	12857	12876	13084	13478	13585	13622	13670	13682
13702	13875	14086	14112	14116	14424	14441	14446	14500	14505	14665	14687
14700	14710	14733	14835	14840	14970	15005	15012	15071	15077	15083	15087
15095	15098	15123	15166	15389	15417	15421	15425	15689	15710	15740	15775
15811	15827	15846	15874	15879	15903	15996	16059	16063	16069	16113	16118
16136	16140	16198	16230	16290	16435	16482	16524	16560	16574	16598	16603
16622	16666	16990	17117	17122	17159	17168	17172	17176	17237	17474	17562
17565	17592	17608	17634	19041	19052	19064	19076	19153	19166	19236	19241
19468	19477	19482	19582	20123	20127	20134	20138	20156	20175	20200	20211
20243	20252	20261	20269	20278	20287	20296	20304	20313	20322	20331	20339
20349	20354	20359	20364	20369	20374	20378	20383	20388	20393	20398	20403
20408	20428	20437	20449	20453	20465	20468	20490	20494	20498	20536	20539
20544	20548	20553	20572	20607	20613	20617	20622	20626	20632	20636	20642
20646	20652	20656	20662	20666	20671	20675	20681	20685	20691	20695	20701
20705	20711	20715	20720	20742	20750	20778	20786	20791	20804	20810	20899
20913	20929	20954	20962	20965	20972	20975	21102	21115	21136	21164	21206
21213	21262	21318	21324	21327	21384	21394	21438	21453	21467	21471	21542
21546	21603	21615	21621	21655	21664	21697	21730	21758	21786	21794	21808
21848	21858	21867	21874	21888	21903	21910	21951	21968	21978	21997	22002
22011	22020	22428	22481	22485	22545	22637	22677	22785	22999	23004	23030
23034	23038	23049	23078	23095	23106	23128	23167	23171	23175	23186	23204
23237	23245	23278	23332	23437	23688	23760	23807	23816	23922	23946	23957
24014	24061	24102	24250	24365	24422	24462	24471	24477	24494	24525	24540
24647	24664	24669	24680	24683	24707	24760	24782	24796	24801	24805	24834
24868	24929	24933	24952	24959	25025	25040	25045	25050	25090	25109	25119
25165	25203	25243	25283	25292	25299	25367	25611	25634	25648	25676	25680
25713	25790	25796	25973	25994	26002	26006	26009	26017	26029	26110	26142
26146	26211	26293	26361	26364	26370	26373	26464	26500	26504	26581	26789
26793	26847	26852	26895	27013	27017	27022	27026	27043	27051	27062	27067
27101	27331	27374	27378	27665	27686	27731	27774	27778	27796	27800	27817
27827	27834	27838	27844	27848	27854	27858	27868	27872	27876	27911	27915
27983	28007	28027	28048	28064	28075	28106	28117	28143	28147	28154	28176
28182	28187	28213	28245	28263	28268	28284	28296	28311	28327	28333	28337

Cross Reference Listing - Field Names and Defined Values

28341	28417	28427	28453	28476	28493	28499	28506	28532	28537	28562	28597
28601	28617	28660	28665	28669	28769	28773	28778	28783	28792	28796	28823
28827	28835	28855	29074	29089	29112	29115	29133	29137	29141	29176	29246
29249	29268	29380	29488	29491	29511	29534	29538	29559	29565	29584	29589
29594	29599	29604	29636	29659	29669	29679	29684	29697	29710	29715	29721
29726	29818	29821	29828	29836	29842	29849	29868	29876	29911	30104	30201
30225	30242	30247	30252	30256	30265	30271	30276	30280	30284	30295	30351
30373	30377	30387	30451	30456	30460	30480	30491	30498	30518	30531	30584
30599	30609	30618	30624	30630	30635	30640	30713	30734	30755	30759	30780
30785	30873	30878	30887	30908	30926	30987	30997	31001	31007	31012	31019
31025	31048	31061	31111	31227	31230	31241	31348	31441	31455	31498	31504
31514	31587	31649	31688	31703	31707	31744	31926	32044	32080	32099	32122
32131	32162	32172	32371	32379	32384	32395	32399	32403	32420	32423	32426
32438	32459	32478	32482	32494	32511	32545	32564	32567	32570	32573	32590
32594	32598	32602	32607	32611	32615	32624	32628	32634	32638	32644	32648
32654	32658	32663	32667	32670	32686	32701	32706	32746	32790	32813	32822
32828	32832	32836	32861	32878	32883	32903	32915	32919	32937	32952	32983
32991	32996	33018	33026	33037	33051	33060	33064	33128	33150	33465	33482
33527	33533	33538	33541	33545	33575	33578	33609	33614	33618	33629	33637
33648	33653	33659	33668	33672	33676	33819	33831	33902	33916	33956	33998
34003	34028	34033	34071	34075	34088	34092	34132	34145	34149	34153	34157
34167	34171	34177	34197	34240	34259	34263	34267	34273	34294	34301	34318
34562	34379	34393	34405	34440	34458	34498	34504	34560	34569	34610	34658
34679	34682	34701	34707	34713	34837	34863	34890	34899	34903	34907	34912
35013	35030	35062	35066	35078	35082	35090	35106	35110	35114	35118	35122
35126	35176	35185	35204	35251	35268	35287	35314	35328	35331	35348	35372
35375	35381	35385	35390	35393	35402	35405	35408	35411	35417	35436	35455
35459	35468	35477	35481	35488	35492	35497	35502	35511	35515	35520	35525
35531	35535	35566	35570	35574	35584	35588	35598	35669	35696	35701	35741
35778	35782	35797	35801	35813	35817	35821	35825	35833	35837	35841	35849
35852	35856	35860	35872	35896	35905	35913	35918	35922	35926	35930	35934
35938	35986	35999	36004	36007	36046	36051	36060	36063	36082	36088	36092
36096	36101	36106	36111	36115	36120	36136	36139	36151	36156	36183	36227
36232	36237	36241	36248	36251	36287	36303	36315	36335	36353	36478	36495
36562	36579	36615	36619	36626	36667	36678	36683	36700	36715	36732	36745
36755	36781	36799	36808	36841	36847	36853	36871	36882	36904	36908	36916
37098	37104	37109	37114	37119	37129	37135	37194	37204	37246	37251	37255
37270	37274	37278	37282	37311	37316	37321	37374	37379	37384	37388	37400
37426	37431	37436	37440	37486	37536	37559	37563	37585	37628	37634	37640
37646	37661	37671	37685	37693	37697	37718	37722	37726	37736	37743	37768
37787	37801	37807	37813	37830	37839	37844	37853	37856	37867	37876	37881
37914	37958	37961	37964	37968	37972	37976	38009	38064	38084	38091	38095
38105	38109	38112	38152	38156	38173	38202	38205	38213	38217	38222	38225
38228	38270	38278	38282	38345	38351	38370	38379	38384	38389	38404	38419
38427	38433	38443	38472	38546	38551	38561	38570	38577	38609	38626	38631
38637	38643	38653	38698	38705	38711	38716	38721	38728	38739	38745	38751
38756	38766	38806	38850	38855	38916	38927	39038	39120	39157	39209	39268
39274	39375	39385	39430	39468	39484	39725	39739	39759	39805	39813	39833
39923	39930	39948	39959	40086	40090	40149	40153	40168	40172	40179	40190
40194	40217	40225	40232	40237	40241	40303	40309	40324	40327	40406	40431
40435	40497	40502	40509	40515	40527	40541	40547	40551	40560	40571	40576
40589	40593	40597	40605	40625	40646	40650	41030	41068	41210	41228	41310
41336	41354	41366	41428	41642	41647	41651	41663	41668	41672	41678	41683
41692	41697	41702	42044	42052	42056	42060	42086	42113	42175	42486	42661
42673	42694	42788	42938	43079	43190	44750	45168	45603	45673	45785	46148

```
                46151  46155  46162  46165  46179  46182  46201  46218  46243  46251  46279  46305
                46313  46344  46352  46367  46386  46391  46402  46407  46416  46427  46432  46441
                46452  46457  46475  46478  46487  46492  46543  46569  46576  46583  46590  46595
                46601  46608  46615  46622  46629  46635  46642  46648  46655  46662  46668  46675
                46682  46689  46696  46716  46724  46729  46739  46744  46747  46763  46780  46807
                46810  46814  46818  46827  46831  46874  46886  46893  46895  46911  46922  46925
                46947  46951  46993  47003  47012  47022  47062  47069  47090  47094  47097  47101
                47108  47112  47135  47139  47153  47157  47161  47165  47168  47172  47176  47181
                47187  47190  47197  47200  47207  47211  47214  47218  47222  47227  47230  47237
                47240  47247  47251  47254  47258  47262  47265  47271  47281  47293  47298  47302
                47336  47356  47369  47378  47393  47401  47405  47411  47415  47422  47430  47469
                47473  47477  47493  47498  47503  47508  47515  47519  47539  47544  47549  47559
                47573  47646  47651  47654  47659  47740  47757  47780  47804  47808  47899  47903
                47935  47942  47947  47982  48032  48084  48087  48090  48093  48096  48099
CON.I           2937 #  5435   5445   5608   5640   5664   5875   6836   7024   21355  29080  30564
                30570  30822  34995  35004  35022  35964  36712  36742  37349  39344  39368  46290
                46977  47006  47049  47056  47323
MISC            2939 #
CLR.FLAGO       2942 #  5342   5601   6132   6268   6556   11472  13347  14435  14450  14827  14978
                15736  15835  15911  16608  17032  17054  17101  17238  17333  20120  20139  21592
                21658  21767  21857  24520  25885  26184  26552  27792  28337  28538  28568  29910
                30210  30248  32371  32862  32920  32982  32995  33050  33721  34298  37195  38286
                42008  45904  45920  45927  45933  45938  45942  45949  45955  45960  45968  46467
                47136  47494  47509  47640
CLR.FLAG1       2943 #  5881   5919   6135   7547   7568   7572   8093   11251  13158  13234  13367
                13473  14454  14513  14867  15357  15361  15537  15543  15679  15725  15753  15759
                15765  15829  15875  15917  16034  16212  16242  16460  16473  16510  16569  16582
                16594  16642  17019  17080  17244  17304  17322  17327  17493  17620  21589  21654
                21998  23245  23902  23908  24062  24366  25625  25709  25857  26614  26775  26819
                35127  35588  37242  38263  46363  46522  47470  47474  47478  47690
CLR.FLAG2       2944 #  5353   6139   6661   6666   6695   6965   7169   7211   7234   7238   8202
                12511  12517  12968  13479  14518  14677  14851  14985  15109  15119  15166  15242
                15824  15887  21811  21915  22638  24201  24205  25868  25872  27012  28664  28782
                29679  29727  30132  30142  30261  30272  30364  30391  30395  30690  30755  30914
                30958  30973  31285  31495  32870  32930  33005  33064  34037  34186  34286  36327
                37559  37877  38562  38654  38739  38745  38751  39161  47936  47947  48012
CLR.FLAG3       2945 #  6142   6595   6867   8097   12161  12228  12525  12532  12989  13361  14037
                14423  14688  14858  14990  15006  15013  15787  15898  15904  17578  20958  21407
                25677  25714  25800  26698  26708  26729  26739  26851  26874  26888  27061  27375
                28341  28764  28852  28836  29481  29762  30266  30280  30285  31300  31534  31745
                31757  34855  35665  36179  36969  36973  36977  37872  38771  38923  39014  39047
                39145  39180  39190  39350  39359  39389  39393  47431  47720
CLR.IPD         2959 #  15706  16277  16520  17487  17508  21983  22016  22024  22046  22093  22642
                22934  23500  23679  23931  24317  24333  24371  24442  24507  25953  25957  26349
                27769  29121  30001  30046  30464  31073  31116  31309  31779  32691  34557
CLR.MMAOINT     2946 #  6152   6221   7872   12359  12704  12710  13664  15399  33076  33081  33086
                40877  40881  40885  45674  48033
CLR.STARTFLG    2947 #  6239   37227  37361  37369  38010  38040  38297  38302  38663  38917  39652
                39947  39958
CLR.TP          2958 #  37188  38292  38643  38932  39175
DEF.SC          2964 #  5904   5910   6148   6814   6886   6889   6973   7021   7027   9600   9608
                9634   9642   20146  20170  20565  20779  20787  20807  21319  28410  28416  28427
                28746  28749  30665  30669  30674  30724  30903  30909  30924  35402  35405  35649
                35658  36000  46151  47397  47411
FLAG.ADR        2962 #  47144  47147
```

Cross Reference Listing - Field Names and Defined Values

```
    FORCE.TB            2961 #  47193   47233
    NOP                 2940 #
    RNUM_2REG           2957 #
    RSB(                2956 #  39760
    SC_14               2967 #  9179    9184    9218    9223    9279    9295    9377    12877   13070   20104   20531
                        28405   28742   35063   35813   36027   36051   36682   36852   39468   39484
    SC_2                2965 #  5897    6145    6159    6255    6708    6811    6883    6969    7036    7094    9169
                        9208    9269    9285    17524   17639   35159   35199   35230   35279   35421   35557   35669
                        37671   46148   47349   47393
    SC_30               2968 #  21127   21132   21244   21249
    SC_6                2966 #  7013    7267    7275    7283    9174    9213    9274    9290    42077
    SET.FLAG0           2949 #  5647    5654    6540    6670    7138    7552    9153    9158    9558    9564    10992
                        11102   11368   11769   12062   12136   13313   13337   14499   14841   15840   16615   17096
                        18197   18216   18273   18751   20099   21971   22007   22580   23556   24499   24532   25728
                        25889   26176   26656   27915   28347   28533   29907   30193   32379   32790   33727   33738
                        34178   34274   37217   37483   38615   38621   39123   39369   41434   42911   42933   42953
                        45899   45909   46551   47499   47504   47630
    SET.FLAG1           2950 #  7377    7481    9400    10739   10746   10874   11225   11327   11363   12155   12580
                        12642   12963   12984   13146   13222   13356   13955   14504   14873   14965   15376   15880
                        16030   16638   17090   17259   17264   17285   17529   17630   18314   20800   21763   21853
                        21993   22532   25629   25862   26619   26771   27177   27182   27186   27257   27361   28618
                        28690   28778   29571   29873   29892   30253   30785   31008   31020   31038   31621   31662
                        31725   32892   32902   32951   32970   33036   33096   35702   37237   41419   41980   41991
                        42027   42427   43052   43074   43092   45915   46556   47680   47809
    SET.FLAG2           2951 #  5430    6689    6959    7161    9382    11629   11645   12958   13950   14696   15127
                        15152   15176   18329   20900   20930   21824   21911   22004   24196   25691   25899   25911
                        25925   25935   26790   26794   26842   27017   27343   29644   30658   30887   31119   31508
                        31719   33530   37115   37120   37130   37136   37487   38156   38164   38206   38279   38578
                        38699   38706   38712   38717   38722   38730   42109   42644   42882   42887   47635   47959
                        48007   48052
    SET.FLAG3           2952 #  6386    6599    9613    9618    12057   12141   12978   13319   13969   14042   16655
                        18371   18678   21054   21063   21165   22720   23436   24541   25681   25821   25832   25342
                        25851   26703   26713   26734   26744   26846   27066   27379   28322   28582   28805   29473
                        30276   30378   31294   31305   31539   33534   36332   37345   37422   38552   38556   43023
                        43028   47560   47685
    SET.FPU             2960 #  16006   17461   21968   22435   22622   22666   22716   23703   25620   26123   26472
                        27668   29924   30104   30201   30351   30480   31469   33553
    SET.MMNOINT         2953 #  6481    7860    8404    8527    8558    8732    8915    8932    8982    9044    9049
                        9329    9341    9501    9506    9726    9736    9742    9749    9755    10752   10766   10776
                        11260   11480   11487   11669   11677   11688   11916   12700   12716   13168   13247   13773
                        13798   14316   14341   14721   14749   14768   14773   14778   14783   15320   15488   16768
                        16809   16824   16[illegible]7   16876   16932   16938   18721   20176   20570   21116   21137   21233
                        21254   21454   21536   21604   21668   21681   21698   21790   21801   21870   21892   21896
                        21988   29650   29655   29711   29716   29722   29991   30042   32812   36308   36614   36766
                        36874   37335   37834   37838   38258   39086   39821   40870   41500   41539
    SET.STACKFLG        2954 #  20966   36288   36336   36342   37205   37247   37252   37331   37427   37981   37985
                        37990   38034   38271   38283   38659   39648
MSRC                    2971 #
    ERRCOD              2984 #  7954    7962    20791   27055   27160   27206   27233   27328   27331   28125   28488
                        28493   28527   28741   28745   28749   36101   36106   36478   36495   36562   36579   37628
                        37634   37640   37646   37651   37697   37708   37718   37726   37743   37804   37819   38005
                        38345   38351   38370   38379   38384   38427   38486   38528   38533   38539   38546   38551
                        38570   38609   38614   38620   38626   38631   38637   38643   38662   38733   38739   38745
                        38751   39120   40086   40090   40149   40153   40168   40172   40190   40194   40431   40435
                        40497   40502   40527   40541   40547   40551   40560   40571   40589   40625   40646   40650
```

Cross Reference Listing - Field Names and Defined Values

	46201	47298	47559	48084	48087	48090	48093	48096	48099			
FPDOFFSET	2985 #	5365	5370	5375	5384	5402	5471	5476	5503	5515	5534	5600
	5687	5723	5939	6127	6163	6228	6273	6972	15689	16198	16230	16435
	17117	17122	17562	20965	21968	22454	22637	22677	22804	23004	23078	23237
	23688	24422	24462	24471	24525	24540	24741	24805	24868	25022	25109	25119
	25299	25648	25790	26293	26581	27026	27665	28417	29074	29924	30036	30104
	30201	30295	30351	30480	30491	30518	31504	32080	32122	33482	34682	34841
	34845	34849	34967	35042	35046	35058	35070	35086	35139	35147	35151	35159
	35163	35172	35190	35195	35199	35208	35217	35230	35234	35243	35260	35279
	35303	35422	35464	35473	35539	35557	35635	35639	35644	35763	35786	35794
	35805	35829	35845	35868	35876	35880	35888	35900	35943	36038	36287	36335
	37246	37251	37419	37426	37431	37657	37666	37704	37722	37736	37787	37867
	38270	38399	38408	38466	38476	38766	38927	39015	39048	39146	39157	40225
	40237	40406	40597	40605	47135	47405						
MA	3009 #											
MDR	3004 #	5388	5392	5553	5693	5741	5760	5776	5806	6756	6782	6790
	7290	8146	8239	8244	8260	8265	8278	8306	8311	8315	8332	8356
	8367	8377	8387	8415	8440	8459	8471	8496	8510	8552	8557	8563
	8589	8595	8601	8619	8635	8652	8678	8684	8690	8708	8726	8731
	8736	8762	8768	8775	8806	8812	8819	8850	8856	8864	8897	8903
	8913	8930	8941	8972	8976	9098	9269	9274	9279	9285	9290	9295
	9300	9305	9376	9665	9697	10683	10692	10702	10808	10816	10825	10829
	10909	10976	11010	11015	11019	11114	11166	11169	11177	11180	11224	11319
	11386	11445	11464	11470	11478	11485	11561	11656	11662	11667	11675	11686
	11696	11703	11768	11780	11933	11982	12033	12040	12154	12165	12227	12744
	12838	12977	12988	13056	13069	13312	13323	13355	13366	13472	13645	13651
	13657	13677	13688	13692	13706	13949	13959	13968	13977	14247	14649	14655
	14767	14772	14777	14782	14947	14952	15083	15098	15108	15308	15476	15693
	15724	15729	15752	15786	15892	15985	15991	16025	16029	16073	16543	16637
	16722	16727	16733	16746	16761	16786	16797	16816	16833	16838	16844	16858
	16905	16968	17018	17070	17075	17080	17085	17095	17232	17269	17441	17448
	17454	17470	17539	17546	18198	18217	18274	18286	18315	18320	18330	18400
	18405	18423	18427	18430	18435	18439	18442	18447	18451	18454	18656	18692
	18698	18714	18726	18914	19009	19021	19047	19060	19070	19082	19086	19108
	19112	19116	19120	19124	19128	19142	19148	19153	19163	19236	19241	19290
	19296	19335	19341	19348	19388	19398	19448	19513	19518	19533	19539	19545
	19553	19564	19571	19698	19719	20098	20103	20164	20168	20175	20183	20207
	20211	20215	20243	20252	20261	20269	20278	20287	20296	20304	20313	20322
	20331	20339	20349	20354	20359	20364	20369	20374	20378	20383	20388	20393
	20398	20403	20408	20539	20560	20564	20572	20579	20584	20589	20594	20764
	20773	21020	21035	21040	21106	21126	21131	21142	21243	21248	21262	21275
	21384	21389	21394	21399	21442	21449	21458	21512	21531	21535	21584	21650
	21662	21712	21715	21723	21758	21782	21848	21864	21881	21972	22008	22418
	22424	22446	22496	22508	22514	22525	22556	22567	22574	22581	22596	22603
	22609	22615	22652	22661	22673	22697	22704	22710	22721	23766	23811	24225
	24231	24256	24262	24456	25095	25098	25601	25625	25629	26088	26095	26099
	26125	26348	26417	26423	26435	26439	26444	26480	26483	26486	26489	29409
	29578	29615	29808	29831	29854	29859	29872	29881	29885	29891	29895	29906
	29919	30083	30089	30094	30099	30125	30130	30174	30180	30185	30189	30220
	30247	30289	30324	30331	30336	30342	30427	30433	30438	30443	30454	31133
	31139	31411	31421	31426	31430	31435	31479	32049	32135	32442	32445	32463
	32466	32485	32498	32501	32514	32525	33465	33473	33500	33503	34372	34387
	34610	34613	34952	34982	35746	36142	36295	36327	36465	36549	36643	36705
	36718	36722	36729	36739	36748	36755	36759	36763	36767	36771	36776	36802
	36812	36875	36885	37226	37872	38167	38292	38354	39215	39274	39338	39346

Cross Reference Listing - Field Names and Defined Values

	39354	39364	39372	39427	39430	39918	39974	40033	40039	40074	40079	40157
	40162	40179	40217	40232	40241	40244	40313	40319	40324	40336	40375	40380
	40420	40425	40609	40639	40642	40901	40929	41024	41035	41042	41062	41073
	41080	41100	41111	41118	41138	41148	41154	41191	41206	41213	41225	41231
	41251	41276	41297	41372	41400	41464	41499	41504	41538	41543	41591	41599
	41607	41619	41626	41647	41651	41668	41672	41708	41712	41746	41755	41801
	41805	41813	41852	41858	41867	41884	41890	41899	41915	41921	41930	41947
	42037	42081	42209	42215	42220	42241	42247	42257	42274	42280	42290	42307
	42313	42322	42356	42438	42447	42486	42511	42515	42565	42570	42619	42624
	42629	42655	42658	42694	42735	42739	42752	42757	42762	42808	42820	42881
	42886	42900	42905	42910	42966	43022	43027	43041	43046	43051	43108	43139
	43181	43213	43225	43230	43235	44348	44352	44361	44386	44401	44409	44430
	44435	44442	44448	44522	44618	44708	44819	44824	44855	44869	44877	44895
	44902	44908	44932	44937	44944	44963	44977	44985	45016	45021	45025	45056
	45251	45274	45279	45308	45323	45331	45336	45354	45359	45368	45392	45407
	45415	45424	45446	45455	45494	45502	45515	45533	45539	45547	45553	45558
	45669	45673	45780	45794	45821	45852	45866	45874	45898	45903	45913	45959
	45967	45982	46008	46012	46057	46065	46076	46084	46410	46435	46460	46771
	46775	46907	46918	47153	47352	47375	47988					
MM.TEMPO	2986 #	8035	27101	27498	27502	27512	27516	27525	27529	27533	27537	35050
	35054	35314	35319	35337	35347	35368	35372	35375	35809	35976	35982	38816
	38822	38850	38855	38897	38905	40212	40221	40333				
PC	3006 #	5454	5458	5462	6138	6152	7041	7045	7116	7129	16005	17466
	17507	19528	19607	19611	19615	19619	19623	19627	19631	19635	19701	19715
	20414	20913	20962	22623	22666	22745	23007	23049	23106	23186	23210	23674
	23760	23807	23816	23875	23969	23974	24010	240 4	24024	24078	24088	24102
	24112	24245	24250	24327	24333	24427	24442	245.0	24688	24697	24715	24815
	24963	25177	25182	25394	26176	26184	27668	28407	28974	30005	30049	31123
	31941	33553	33575	33956	34553	34594	34599	35593	35616	36527	36715	36841
	37440	37755	37763	37768	37776	37908	38492	39011	39078	39090	39209	39268
	39644	39755	39884	40255	40368	40390	41030	41068	41106	41143	41179	41263
	41489	41528	41603	41718	41722	41766	41863	41895	41926	42195	42252	42285
	42318	42379	42388	42541	42666	42748	42788	42814	42896	42938	42958	43037
	43079	43097	43147	43190	44550	44750	44798	44996	4500	45066	47396	
PCBACK	3008 #	6122	16010	16293	17581	17613	22748	22788	25253	25296	27671	27768
	28846	29120	31244	31853	31901	31944	32034	32692	34604	34616	34661	37344
	37436	38105	38806	39043	39082							
PSHADD	2998 #	39334	39341	42603	42829	42846	42976	42992	43117	43131	43167	44425
	44455	44528	44543	44624	44639	44718	44728	44890	44914	45032	45039	45137
	45148	45246	45524	45563	45773	45801						
PSHSUB	2999 #	8455	8467	19640	19723	20191	21153	21159	42403	42558	42580	42773
	42777	42783	42921	42926	42932	43062	43067	43073	44368	44484	44579	44669
	44758	44836	45011	45074	45186	45291	45374	45461	45608	45718	45833	45921
	46024											
READRBS	3000 #	38961										
RNUM_WBUS	2994 #	6170	6239	6269	6760	6879	7584	7588	8167	8177	9498	9510
	14762	14789	14794	16926	16942	20613	20622	20632	20642	20652	20662	20671
	20681	20691	20701	20711	20720	21267	27062	27067	27288	27838	27848	27854
	27858	28038	28159	28176	29380	29415	29636	29649	29654	32371	32379	32517
	32682	32790	32867	32870	34867	34956	35488	36227	36232	36237	36241	36258
	36687	36696	36847	36865	39493	39507	46162	46182	46344	46352	46367	46386
	46402	46427	46452	46487	46543	46550	46555	46716	46724	46729	46739	46747
	46780	47062	47069									
S[98	2987 #	5973	20899	20929	35247	35774	36311	37098	37114	37119	37129	37135
	37876	38091	38112	38163	38213	38228	38278	38389	38404	38433	38443	38472

Cross Reference Listing - Field Names and Defined Values

	38561	38577	38653	38698	38705	38711	38716	38721	38728	38756		
SISR	2988 #	5903	35034	35038	35287	35298	35790	37486	37509	39385		
TB	3010 #	36125										
TEMP.R	2990 #	8173	27086	27937	27949	27989	28014	28171	29412	29669	29674	29710
	29715	32490	32628	32634	32638	32644	32654	32670	32686	32701	32706	32843
	32847	32857	32861	32878	32883	32903	32915	32919	32925	32929	32937	32941
	32947	32952	32963	32968	32978	32983	32991	32996	33001	33006	33011	33014
	33026	33030	33037	33040	33046	33051	33155	46364	46472	46478	46482	46495
	46720	47066	47097									
TEMP.R+1	2991 #	29659	29663									
TEMP0	2973 #	5327	5333	5438	5441	5448	5451	5526	5617	5635	5754	6502
	6585	6585	6603	6603	6607	6647	6685	6861	7643	7670	7670	7694
	7814	7823	8150	8291	8296	8300	8320	9124	9309	9318	9327	9387
	9505	9558	9569	12336	12352	12358	12378	12633	12636	12741	12891	13609
	13663	13673	14228	14493	14508	14660	15340	15836	15841	16287	16290	17316
	17521	17527	17534	17629	18729	19309	19314	19404	19643	19726	20197	20200
	20203	20471	20745	21099	21102	21111	21201	21206	21213	21216	21310	21314
	21318	21324	21327	21330	22781	22785	23137	23204	23214	23288	23332	23342
	23680	23932	24647	24650	25302	25312	25325	27117	27121	27125	27129	27431
	27441	27446	27727	27911	28069	28411	28427	28453	28506	28510	28510	28532
	28537	28593	28597	28606	28617	28622	28632	28636	28640	28764	28769	28778
	28783	28809	28823	28827	28831	28835	28849	28855	28858	28865	28956	28978
	29083	29086	29089	29092	29109	29112	29115	29124	29159	29162	29227	29272
	29275	29334	29338	29346	29349	29369	29462	29473	29478	29494	29499	29643
	29754	29758	29876	30826	31163	31172	31175	32037	32197	32214	32554	32727
	33128	33131	33134	33137	33562	33609	33614	33637	33648	34710	34713	34758
	35813	36051	36128	36248	36683	36700	36745	36781	36785	36808	36853	36871
	36882	36888	36911	36914	36965	38952	38970	38974	41257	41268	41279	41612
	41655	41726	41761	41770	41781	41990	42002	42007	42014	42090	42113	42171
	42175	42397	42489	42492	42507	42550	42661	42670	42673	43032	44567	44596
	45062	45093	45107	45114	45168	45173	45211	45226	45233	45480	45594	45603
	45626	45641	45648	45658	45707	45736	45751	45758	45943	45977	45990	46043
	46279	46299	46566	46810	46818	46822	46869	46980	46996	47570	47573	47947
	47982											
TEMP1	2974 #	5530	5539	5662	5681	5731	5784	6580	6600	6617	6678	6743
	6764	6800	6828	6839	6864	6871	6875	6883	6889	6922	7051	7106
	7110	7113	7183	7190	7667	7781	7809	7812	8403	8501	8606	8695
	8780	8824	8869	8985	9027	9043	9048	9055	9063	9323	9339	9476
	9747	9808	9812	9816	9820	9825	9843	10738	10745	10751	10763	10773
	10867	10878	10914	11092	11101	11110	11121	11129	11550	11639	11760	11787
	11793	11798	11824	12347	12389	12410	12738	13151	13227	13330	13336	13371
	13448	13460	13761	13780	13861	13981	14076	14107	14304	14323	14418	14450
	14935	14959	16039	16225	16238	16246	16419	16538	16555	16654	17195	17199
	17203	17210	17214	17218	17221	17224	17311	17477	18240	18244	18363	18367
	18371	18395	18659	18671	18678	19278	19588	20950	20954	21115	21136	21259
	21350	21358	21361	21411	21434	21438	21453	21482	21588	21612	21719	21771
	21819	21823	21910	21978	21994	21997	22002	22011	22015	22020	22057	22092
	22627	22715	24446	24515	24560	24789	24876	25102	25262	25288	25292	25307
	25318	25379	25382	25608	25611	26104	26135	26180	26211	26240	26283	26317
	26451	27195	27435	27449	27465	27692	27868	27915	28151	28476	28499	28558
	28567	28573	29424	29427	29559	29773	29776	29779	30574	30584	30599	30624
	30640	30648	30785	30829	30832	30836	30848	30883	30892	31039	31043	31048
	31108	31119	31214	31227	31415	31905	31911	31932	32089	32111	32200	32607
	32730	33164	33229	33522	34167	34171	34301	34462	34501	34542	34547	34550
	34619	34676	34679	34837	34863	34886	34890	34899	34903	34964	34974	35026

Cross Reference Listing - Field Names and Defined Values

	35030	35074	35094	35118	35168	35176	35179	35213	35221	35251	35256	35293
	35308	35342	35356	35363	35398	35430	35433	35440	35445	35450	35468	35481
	35551	35566	35570	35574	35579	35621	35759	35892	35896	35948	35951	35957
	36010	36012	36015	36018	36136	36146	36160	36482	36499	36503	36508	36566
	36583	36587	36618	36622	36732	36736	36791	36793	36796	36799	36895	36904
	36908	36916	36919	36922	39111	40857	40877	40881	40885	41301	41305	41321
	41327	41332	41346	41377	41404	41408	41412	41418	41423	41433	41438	41480
	41485	41493	41519	41524	41532	41973	41979	41995	42019	42076	42107	42149
	42393	42426	42643	42706	42719	45442	45589	45677	45702	46572	46604	46638
	46671	46807	46827	46838	46878	46983	47161	47172	47207	47218	47247	47258
	47278	47281	47326	47336	47341	47369	47613	47619	47629	47634	47639	47646
	47651	47654	47666	47670	47679	47689	47695					
TEMP10	2983 #	6155	6166	6752	6803	6831	7271	7279	7287	7923	8194	8205
	14052	16259	16440	16452	16456	22502	22531	22536	22561	22939	23225	23230
	23234	23357	23479	23487	23514	23569	23573	23693	23897	23903	24307	24519
	24536	24557	24575	24640	24655	24757	24768	24925	24969	25056	25061	25069
	25210	25213	25401	25405	27051	27164	27216	27229	27325	28128	28484	28524
	28733	28737	29834	29842	29911	31493	31508	31533	31538	31548	31553	31574
	33198	34253	34279	34290	34304	34349	34405	34408	37265	37287	37326	39222
TEMP2	2975 #	5421	7647	7658	7658	7698	7795	7804	9071	9415	9424	9669
	9686	9693	9702	9710	9716	9720	9735	9741	9763	9773	9781	9786
	9791	9795	9799	9829	10981	10991	10995	11001	11135	11139	11233	11243
	11258	11334	11340	11351	11358	11367	11373	11383	11440	11570	11629	11633
	11650	11880	11888	11896	11907	11913	11977	12046	12051	12061	12069	12127
	12135	12145	12160	12232	12368	12404	12524	12531	12545	12549	12575	12590
	12598	12603	12833	12863	12887	12949	12957	12967	13052	13089	13156	13165
	13232	13244	13453	13766	13771	13785	13796	13881	14097	14309	14314	14328
	14339	14424	14639	14676	14683	14720	14748	14814	14819	14940	14991	15136
	15169	15187	15304	15318	15351	15366	15393	15398	15486	15684	15770	15792
	15805	15811	15855	15860	15909	16447	16491	16496	16505	16578	16581	16628
	16986	17025	17274	17550	18221	18278	18290	18334	18378	18382	18459	18464
	18469	18482	18486	18497	18501	18512	18516	19575	19578	20127	20138	20597
	20603	20724	20750	20756	21164	21464	21471	21521	21549	21557	21618	21786
	21794	21808	21888	21916	21919	22054	22457	22772	22916	23684	23743	23747
	23751	23791	23796	23857	23861	23865	23936	24531	24551	24760	24763	24774
	24929	24933	24936	24944	24948	24988	24992	24997	25040	25064	25075	25080
	26084	26123	26223	26255	26277	26472	26827	26835	26838	26841	27423	27427
	27454	27494	27506	27520	27872	28143	28147	28401	28792	28796	28982	29391
	29421	29524	29554	29565	29570	29574	29639	29765	29768	29814	29899	29902
	29937	29949	30001	30011	30029	30033	30447	30451	31126	31233	31469	31908
	31914	32115	32118	32203	32611	32733	33232	33487	33506	33514	33518	33527
	33541	33545	33549	34556	34607	34654	34658	34872	34876	35265	35268	35271
	35275	35284	35328	35334	35353	35359	35387	35390	35393	35402	35455	35543
	35546	35770	35778	35782	35797	35801	35908	35954	35961	35967	35971	35986
	35991	36072	36076	36081	36087	36096	36111	36115	36120	36131	36139	36151
	36156	36166	36299	36630	36635	39033	40854	41362	41366	41384	42041	42044
	42048	42052	42056	42060	46579	46611	46645	46678	47719	47736	47744	47793
	47814	47862	47881	47991	48017	48021	48036	48048				
TEMP3	2976 #	6660	6665	6892	6900	6907	6914	6925	6935	6956	6959	6964
	6990	7017	7125	7180	7187	7205	7223	7655	7786	7789	7792	9381
	9676	9681	9706	9770	9805	11555	11644	11928	12585	12594	12606	12615
	13590	13602	14048	14058	14069	14512	14728	14803	14807	15000	15146	15151
	15175	15182	15212	15220	15493	15519	16501	16622	17278	17321	17326	17460
	18202	18207	18226	18230	18235	18734	19524	19557	19593	20155	20219	20233
	20465	20486	20490	20494	20498	20530	20536	20569	21467	21539	21542	21553

Cross Reference Listing - Field Names and Defined Values

	22541	24824	24831	24880	24883	25676	25680	25800	25884	25890	26168	26321
	26603	26699	26704	26709	26714	26730	26735	26740	26745	26789	26793	26847
	26852	27029	27374	27378	27696	27731	27774	27778	27876	27970	27983	28007
	28021	28048	28064	28182	28187	28213	28311	28971	29623	30346	30456	30460
	30535	31181	31235	31336	31591	31629	31678	31688	31703	31707	31871	31874
	31881	31884	31887	31891	31923	31926	31929	31938	31947	32096	32099	32139
	32143	32151	32155	32158	32172	32176	32206	32234	32251	32270	32285	32288
	32420	32423	32476	32438	32459	32478	32494	32535	32545	32573	32615	32737
	32746	33150	33175	33207	33235	33492	33496	33525	33533	33538	33578	33618
	33629	33632	33653	33659	33663	33668	33672	33676	33813	33907	33916	33992
	34003	34008	34028	34071	34075	34088	34092	34132	34149	34153	34157	34240
	34259	34263	34267	34294	34504	34623	34692	34701	35062	35411	35417	35427
	35999	36004	36046	36360	36469	36553	39074	41341	47082	47090	47108	47731
	47740	47776	47797	47828	47844	47859	47877	47892				
TEMP4	2977 #	7689	8909	9018	9038	9147	9152	9157	9162	12693	13635	13640
	14434	15710	15922	15996	16013	16515	16524	16528	16990	17237	17481	17619
	17625	17634	17638	20428	20437	20453	21030	21475	21546	21597	21607	21655
	21668	21673	21681	21686	21701	21730	21768	21790	21813	21858	21870	21892
	21903	21906	21951	21958	21988	22046	22751	22991	23060	23064	23562	24718
	24808	24819	24828	24843	24847	24859	25817	25821	25895	25900	26475	26615
	26620	26757	26808	27032	27700	27711	27800	27813	27817	27820	27884	27919
	28240	28245	28248	28257	28322	28327	28341	28562	28582	28582	28588	28588
	28601	28773	29100	29129	29133	29141	29145	29246	29627	30193	30193	30589
	30606	30840	31178	31238	31241	31595	31626	31682	32237	32280	32564	33171
	33215	33530	35996	36043	36068	36267	36271	36275	36279	36283	36293	36303
	36331	36345	37334	37352	38267	38289	39086	41447	41451	41952	46586	46618
	46651	46685	47684	47832	47848	47856	47873	47889				
TEMP5	2978 #	5398	5408	5417	5590	5655	5659	5675	6135	6148	6159	6224
	6255	6264	6280	6284	6292	6298	6306	6312	6320	6326	6334	6340
	6348	6354	6362	6368	6376	6382	6390	6396	6404	6410	6418	6424
	6432	6463	6470	6477	6486	6527	6535	6556	6590	6612	6620	6857
	6931	7176	7226	7263	7267	7275	7283	7309	7313	7321	7329	7337
	7344	7352	7360	7368	7382	7404	7409	7432	7437	7453	7458	7475
	7498	7503	7520	7525	7542	7547	7556	7568	7577	7594	7614	7626
	7634	7638	7651	7663	7675	7686	7705	7713	7717	7740	7745	7756
	7761	7768	7799	7818	7828	7832	7849	7853	7860	7871	7876	7907
	7911	7915	7967	7973	7985	8018	8030	8040	8045	8050	8057	8062
	8067	8072	8075	8079	8093	8186	8190	9453	9464	9470	9481	11326
	11344	11379	11811	12221	12236	12554	12559	12619	12626	12843	12857	13061
	13084	13466	13478	13615	13618	13870	13875	14429	14441	14687	14700	14710
	14733	14825	14976	15005	15012	15071	15077	15087	15093	15102	15130	15205
	15238	15816	15822	15827	15846	15874	15879	15886	15903	16472	16482	16588
	16666	18651	18702	18707	18723	18746	18750	18754	18757	18761	18769	18772
	20144	20186	20778	20786	20796	20800	20804	20810	21600	21603	21615	21705
	21733	22761	22764	23030	23034	23038	23095	23167	23171	23175	23258	23306
	23310	23473	24632	24747	24752	24952	24956	25030	25035	25165	25168	25196
	25203	25243	25249	25276	25283	25827	25832	25906	25912	26496	26623	26760
	26811	27035	27682	27737	28302	28306	28347	28554	28669	28942	29103	29137
	29154	29176	29249	29268	29469	29514	29529	29542	29549	29604	29981	29986
	30595	30689	30695	30699	30852	30856	30860	30864	31128	31218	31221	31270
	31309	31312	31598	31617	31622	31685	32240	32277	32567	33073	33077	33167
	33218	33715	33731	33858	33862	33898	34020	34067	34084	34145	34192	34197
	34309	34314	34318	34362	34393	34435	34440	34458	34494	34498	34736	34740
	34744	36030	36034	36060	36065	37379	37388	37491	37512	38084	38087	38098
	38101	38205	38209	38219	38222	38225	39024	39038	39397	42118	42124	42133

Cross Reference Listing - Field Names and Defined Values

	42143	46151	46155	47393	47411	47422	47749	47757	47780	47788	47835	47851
	47886											
TEMP6	2979 #	5491	5495	5499	5507	5977	6288	6302	6316	6330	6344	6358
	6372	6386	6400	6414	6482	6498	6507	6512	6516	6541	6544	6825
	7192	7207	7317	7325	7333	7341	7348	7356	7364	7372	7376	7386
	7401	7413	7429	7 71	7450	7462	7472	7480	7485	7495	7507	7517
	7529	7539	7560	7564	7581	7591	7749	7753	7978	8198	9022	14073
	14082	14455	14470	14474	18718	20110	20116	20475	20518	20607	20617	20626
	20636	20646	20656	20666	20675	20685	20695	20705	20715	22428	22481	22485
	22545	22999	23245	23437	23922	23946	24494	24664	24672	24680	24683	24692
	24707	24782	24796	24801	24834	24959	24973	24979	25015	25025	25045	25050
	25090	25371	25837	25842	25919	25926	26171	26216	26280	26341	26626	26763
	26814	27038	28217	28284	28945	28962	29106	29166	29180	29184	29218	29223
	29234	29253	29257	29280	29284	29360	29402	29481	29485	29488	29491	29511
	29519	29534	29538	29545	29584	29589	29594	29599	29818	30210	30225	30229
	30242	30252	30256	30265	30271	30284	30355	30369	30505	30509	30708	30741
	30745	30768	30913	30918	30923	30957	30962	30972	30977	31012	31061	31068
	31587	31603	31649	31691	32243	32274	32570	33082	33221	3378?	33805	33835
	33893	33902	33945	34754	36026	36056	36063	38482	39162			
TEMP7	2980 #	6260	6746	6817	6903	6910	6917	7036	7054	7058	7070	7082
	7094	7165	7169	7215	7230	7234	7238	7895	7899	7919	8087	8127
	8131	8135	8139	8143	12395	13585	13622	13682	13698	14223	14238	14241
	14258	17587	17592	17601	17608	20424	21005	21045	21049	22955	23500	24629
	24660	24669	24735	26192	26252	26337	27022	27796	27941	27945	28106	28117
	28703	28725	29821	29828	29846	29849	29868	30118	30215	30468	31441	31578
	31634	32162	32210	32247	32624	32813	33119	34666	34669	34685	34724	34728
	34732	35669	36183	36254	36263	36307	36315	36625	36639	37360	37368	37671
	38149	38152	46874									
TEMP8	2981 #	6651	6748	6769	6867	7062	7074	7086	7098	7119	7392	7396
	7398	7419	7423	7426	7444	7447	7465	7469	7721	8097	14705	14835
	14840	15385	15389	15417	15421	15425	16414	16425	16460	16466	22801	22969
	22972	22985	22994	24477	24675	24678	24701	24704	24721	24785	24792	25367
	25639	25643	25655	25686	25858	25863	26150	26237	26274	26509	26578	26607
	27013	27017	27172	27292	28134	28457	28461	28713	28717	30495	30567	30679
	30731	30771	31460	31517	31860	32040	32044	32063	32067	32093	32590	32594
	32598	32602	32648	32658	32663	32667	32719	32723	33182	33559	33720	33726
	33737	33809	33998	34033	34036	34177	34185	34273	34285	34297	34560	34569
	34704	34707	36350	36353	37104	37143	37181	37187	37194	37204	37270	37274
	37278	37282	37311	37316	37321	37340	37384	37396	37413	37422	37536	37559
	37585	38064	38146	38173	38028	46218	46238	46276	46286	46308	46521	46528
	46533	46538	46560	46593	46625	46658	46692	47493	47498	47503	47508	47865
	47935	47942	47952	47963	47969	47978	48043					
TEMP9	2982 #	6174	6177	6772	6811	6895	7065	7077	7089	7101	7122	7294
	7488	7492	7510	7514	7532	7536	8103	14038	14043	14062	15299	15313
	15362	15371	15375	15471	15480	16255	16428	22450	23128	23133	23141	23278
	23283	23292	24635	24744	26130	26153	26515	26588	27007	27043	27168	27226
	27315	28131	28469	28522	28721	28729	31482	31522	31714	32054	32058	32071
	32131	33190	37199	37230	37235	37240	37294	37298	37302	37306	37690	37693
	37827	37830	37841	37844	37984	38033	38039	38262	38275	38296	38301	39066
	39070	39381	39403	39407	39411	39415	39419	39441	39445	39449	39452	39462
	39474	46212	46230	46235								
VA	3005 #	5698	5712	5715	5823	5997	7219	8121	8640	9395	9494	9778
	12579	12641	14757	14866	15706	15714	16052	16124	16128	16140	16277	16520
	16532	16910	16922	16973	17110	17155	17159	17164	17168	17172	17176	19064
	20160	20462	20556	20730	20734	20738	20742	20767	20814	20819	21122	21221

Cross Reference Listing - Field Names and Defined Values

		21228	21239	21517	21664	21697	21762	21852	23493	23669	23928	24433	24489
		24506	24863	28154	28333	28337	28394	28660	28665	28953	29679	29684	29688
		29697	29721	29726	29730	30009	30630	30635	30713	30755	30759	30873	30878
		30887	30908	30926	30967	31333	31348	32384	32395	32399	32403	32482	32511
		32794	32822	32828	32832	32836	33018	35520	35584	35598	35612	35624	35701
		36474	36486	36491	36558	36570	36575	36678	37255	37678	37747	37810	37816
		37911	38357	38916	39692	39725	39739	39759	39801	39805	39809	39813	39833
		39874	39908	39948	39959	39964	39996	40054	40140	40249	40297	40300	40309
		40327	40330	40365	40398	40441	40482	40487	40492	40506	40515	40531	40537
		40566	40576	40584	40593	40621	40630	40655	40692	40701	40705	40755	40762
		40766	40814	40821	40862	40873	42464	42597	42854	43000	43103	43177	43185
		44533	44629	44713	44991	45120	45785	46179	46951	47018	47193	47204	47233
		47244	47808										
WBUS_RNUM		2995 #	5380	7132	9399	9485	14716	14744	14857	14872	16915	16978	
WB_RBSP		3001 #	22098	37862	38761	38922	39152						
WDR		3011 #	39722	39736									
XB.PC_PC+1		3007 #	5467	5671	7031	7198	15406	17499	18970	18975	18980	18985	18990
		18995	19000	19004	19092	19096	19183	19187	19248	19253	19283	19329	19381
		19444	19658	19663	19668	19673	19678	19683	19688	19692	20480	23017	23021
		23090	23154	23158	23449	23453	23464	23468	23730	23734	23782	23786	23844
		23848	23978	23984	23988	24039	24047	24073	24082	24128	24135	24195	24313
		24342	26302	27752	27979	28003	28060	28686	28691	30205	30484	30579	30644
		30664	30668	30673	30723	30896	30902	30936	30941	30946	30982	30990	31032
		33146	33571	33623	42767	42792	42797	42859	42916	42942	42947	43057	43083
		43087	43152	43197	43202	43207	44357	44372	44377	44382	44391	44396	44476
		44488	44492	44496	44504	44507	44571	44583	44588	44592	44600	44603	44661
		44673	44678	44682	44690	44693	44762	44766	44770	44778	44781	44840	44845
		44850	44859	44864	44949	44954	44959	44968	44972	45006	45078	45083	45088
		45096	45101	45142	45192	45199	45205	45215	45220	45295	45300	45305	45313
		45318	45364	45378	45383	45388	45397	45402	45451	45465	45470	45475	45484
		45489	45529	45598	45611	45616	45621	45630	45635	45682	45721	45726	45731
		45740	45745	45817	45837	45842	45847	45856	45861	45908	45925	45931	45936
		45947	45953	45987	46028	46033	46038	46047	46052	46882	46889	46989	46999
		47008	47015										
MUX		3014 #											
	D.Q1	3025 #	15126	15201	15328	15333	15502	15508	15525	15530	15535	15541	24485
		46215	46276	46301									
	D.Q2	3026 #											
	D.R1	3023 #	8177	8919	8936	8946	9058	9066	9489	9573	9584	13881	14228
		14241	14991	16491	18521	18524	18711	19321	19409	19413	19417	22461	22969
		23053	23907	25660	25717	25806	25868	25872	26428	26593	26645	26876	26899
		27048	27288	27704	28090	28397	29213	29355	29930	29934	31527	31752	32375
		32517	32550	32586	32682	32867	32870	37482	38393	38437	38770	39179	39189
		41957	41962	41969	41990	42002	42007	42153	42864	43005	43125	43157	46246
		46735											
	D.R2	3024 #	46226										
	D.S	3027 #	5398	5408	5590	5614	5625	5630	5749	5796	5947	6006	6779
		6787	6795	7062	7065	7074	7077	7086	7089	7098	7101	7876	9514
		9518	9522	9526	9760	11824	11888	12331	12341	12371	12575	12863	13670
		14212	14676	15698	15792	16118	18249	18765	19468	19477	19482	19582	20123
		20127	20134	20138	20190	20200	21025	21621	21775	21780	21804	21867	21874
		21878	21899	22467	22473	22477	22630	22683	22687	22725	22952	23497	23699
		23943	23957	23997	24004	24019	24061	24092	24107	24365	24450	24782	26110
		26464	27071	27075	27823	27827	27834	27844	27889	27964	27995	28027	28075
		28166	30498	30734	30780	30987	30997	31001	31007	31019	31025	31111	31455

Cross Reference Listing - Field Names and Defined Values

	33060	33064	33819	33831	33835	34011	35185	35204	35328	35331	35334	35436
	35531	35535	35584	35598	35696	35701	36072	36151	36156	36678	37685	39375
	41354	42026	42032	42158	42163	42838	42984	46182	46238	46367	46386	46402
	46427	46452	46487	46543	46720	46729	46747	46780	47066	47069	47814	
M.Q1	3017 #	5693	5741	8239	8260	8557	8563	8595	8601	8652	8684	8690
	8731	8736	8768	8775	8812	8819	8856	8864	8913	8930	8941	9043
	9048	9055	9063	9318	9327	9505	9558	9569	13615	14223	14238	14258
	14952	15102	15130	15205	16128	19082	19086	19108	19112	19116	19120	19124
	19128	19290	19296	19404	19513	19553	21035	21040	21597	21673	21686	21958
	23473	24557	24672	24718	25253	25296	30595	30689	30695	32794	36759	40330
	41179	41599	41607	41626	41852	41867	41884	41899	41915	42037	42209	42215
	42220	42241	42274	42307	46235	46308	46410	46435	46460	46528	46533	46538
	46593	47375										
M.Q2	3018 #	46286	46482	46572	46579	46586						
M.R1	3015 #	5421	5635	5776	5806	6122	6585	6603	6764	6839	6883	7045
	7129	7190	7203	7223	7658	7670	7799	7818	7853	7978	8244	8265
	8552	8589	8635	8640	8678	8726	8762	8806	8850	9038	9481	9816
	9820	11010	11015	11019	11129	11135	11139	11169	11180	11344	11351	11358
	11367	11464	11470	11633	11650	11656	11662	12221	12352	12358	12368	12524
	12531	12585	12598	12626	12738	12741	12744	12843	13061	13466	13590	13602
	13635	13645	13651	13657	13677	13698	13870	14247	14429	14450	14455	14470
	14474	14683	14705	14772	14825	14976	15340	15714	15805	15816	15822	15855
	15860	15886	15922	15991	16010	16073	16293	16472	16528	16581	17075	17521
	17539	17581	17613	18207	18226	18230	18235	18320	18405	18656	18692	18718
	19021	19142	19148	19163	19309	19314	19335	19341	19398	19518	19528	19557
	19588	19593	19719	20098	20215	20486	20594	20756	20796	21020	21045	21049
	21111	21216	21310	21330	21358	21389	21399	21475	21553	21557	21588	21607
	21612	21662	21668	21681	21701	21712	21723	21733	21768	21782	21790	21813
	21864	21870	21881	21892	21906	21916	21919	21988	22046	22536	22748	22788
	22801	22804	22916	22972	22994	23060	23064	23141	23292	23468	23487	23514
	23562	23811	23903	23928	23984	24047	24073	24135	24427	24519	24536	24632
	24650	24678	24704	24763	24831	24880	24944	24956	24979	25064	25168	25249
	25608	25858	25863	25884	25890	25895	25900	25906	25912	25919	25926	26104
	26171	26180	26192	26252	26280	26317	26321	26337	26341	26451	26578	26588
	26699	26704	26709	26714	26730	26735	26740	26745	26757	26760	26763	26808
	26811	26814	26827	26835	26838	26841	27160	27164	27168	27172	27195	27206
	27216	27226	27229	27233	27431	27435	27446	27449	27454	27516	27529	27537
	27671	27700	27884	28248	28257	28322	28394	28407	28457	28461	28469	28484
	28488	28510	28522	28524	28527	28567	28582	28588	28858	28953	28956	28974
	29086	29092	29473	29478	29499	29514	29519	29529	29643	29688	29730	29773
	29779	29899	29906	29919	29937	30118	30193	30215	30447	30454	30468	30495
	30509	30574	30589	30679	30699	30731	30771	30829	30840	30883	30892	31108
	31244	31336	31415	31508	31517	31533	31538	31553	31678	31682	31685	31691
	31853	31874	31884	31901	31905	31908	31923	31929	31938	31944	32063	32067
	33073	33077	33137	33496	33506	33632	33805	33893	33945	34547	34550	34556
	34604	34616	34661	35034	35168	35284	35298	35337	35427	35593	35967	36030
	36142	36307	36311	36482	36486	36499	36503	36508	36566	36570	36583	36587
	36643	36722	36748	36785	36888	37352	37396	37413	37422	37509	37763	38267
	38289	39028	39043	39082	39086	39346	39372	39427	39755	39884	40255	40368
	40877	40881	40885	41106	41191	41263	41297	41372	41400	41408	41464	41485
	41489	41493	41499	41504	41524	41528	41532	41538	41543	41612	41708	41712
	41718	41722	41858	41890	41921	41930	42048	42143	42195	42247	42257	42280
	42290	42313	42322	42397	42464	42511	42515	42624	42706	42748	42814	42854
	42859	42896	42958	43000	43037	43097	43147	43177	43185	44357	44382	44448
	44476	44496	44567	44571	44592	44596	44618	44629	44661	44682	44770	44850

Cross Reference Listing - Field Names and Defined Values

	44908	44991	44996	45120	45205	45251	45305	45364	45388	45475	45529	45558
	45621	45682	45731	45794	45847	45936	45987	46038	46604	46611	46618	46625
	46638	46645	46651	46658	46671	46678	46685	46692	46822	46838	46907	46918
	47719	47731	47744	47856	47859	47862	47865	47952	48017	48021		
M.R2	3016 #	11555	11644	14076	14097	15000	15836	15841				
M.S	3019 #	5333	5370	5375	5388	5392	5417	5441	5451	5454	5458	5462
	5495	5499	5507	5539	5655	5659	5671	5675	5681	5698	5712	5715
	5760	5784	5823	5997	6138	6152	6177	6228	6280	6284	6298	6312
	6326	6340	6354	6368	6382	6396	6410	6424	6463	6470	6477	6498
	6507	6512	6516	6541	6544	6590	6607	6647	6748	6756	6772	6817
	6828	6861	6867	6871	6875	6889	6892	6895	6900	6903	6907	6910
	6914	6917	6922	6925	6935	6959	6990	7017	7041	7051	7054	7058
	7070	7082	7106	7110	7116	7119	7122	7125	7165	7180	7183	7187
	7207	7219	7226	7230	7271	7290	7294	7313	7317	7321	7325	7329
	7333	7337	7341	7344	7348	7352	7356	7360	7364	7368	7372	7376
	7386	7401	7413	7429	7441	7450	7462	7472	7480	7485	7495	7507
	7517	7529	7539	7560	7564	7581	7591	7614	7626	7634	7643	7647
	7675	7689	7694	7698	7713	7717	7721	7745	7749	7753	7789	7795
	7809	7814	7828	7832	7849	7907	7911	7915	7923	7962	7973	8030
	8057	8062	8067	8072	8075	8079	8121	8143	8198	8278	8306	8356
	8367	8377	8403	8415	8440	8459	8471	8501	8606	8619	8695	8708
	8780	8824	8869	8985	9071	9098	9339	9376	9395	9415	9424	9494
	9676	9681	9720	9770	9778	9791	9805	9812	9829	10692	10702	10751
	10763	10773	10816	10825	10829	10867	10878	11101	11110	11258	11326	11340
	11478	11485	11667	11675	11686	11696	11703	11768	11780	11787	11798	11811
	11896	11907	11913	11928	11933	12033	12040	12227	12236	12336	12347	12378
	12395	12404	12549	12554	12559	12579	12594	12619	12636	12641	12693	12857
	13084	13165	13244	13472	13478	13585	13622	13640	13663	13673	13682	13771
	13796	13875	14038	14043	14048	14058	14069	14107	14314	14339	14434	14441
	14508	14660	14687	14700	14710	14720	14733	14748	14757	14767	14777	14782
	14802	14807	14814	14819	14866	15005	15012	15071	15077	15083	15087	15093
	15098	15151	15175	15182	15318	15389	15417	15421	15425	15486	15493	15693
	15706	15710	15729	15811	15827	15874	15879	15892	15903	15909	15996	16005
	16013	16029	16052	16124	16140	16277	16290	16452	16460	16466	16482	16496
	16501	16515	16520	16524	16532	16538	16543	16622	16628	16637	16666	16910
	16922	16973	16990	17025	17110	17155	17159	17164	17168	17172	17176	17232
	17237	17274	17278	17311	17316	17441	17448	17460	17466	17470	17477	17481
	17507	17534	17592	17608	17634	18221	18240	18244	18278	18400	18427	18439
	18451	18726	18769	18772	19009	19064	19153	19236	19241	19348	19388	19448
	19533	19539	19545	19607	19611	19615	19619	19623	19627	19631	19635	19643
	19698	19701	19715	19726	20160	20164	20175	20197	20203	20211	20243	20252
	20261	20269	20278	20287	20296	20304	20313	20322	20331	20339	20349	20354
	20359	20364	20369	20374	20378	20383	20388	20393	20398	20403	20408	20414
	20424	20428	20437	20453	20462	20465	20471	20475	20490	20494	20498	20536
	20539	20556	20569	20572	20579	20584	20589	20603	20607	20617	20626	20636
	20646	20656	20666	20675	20685	20695	20705	20715	20724	20730	20734	20738
	20742	20745	20750	20767	20773	20778	20786	20791	20804	20810	20814	20819
	20899	20913	20929	20954	20962	21102	21122	21142	21164	21206	21213	21221
	21228	21239	21275	21318	21324	21327	21361	21384	21394	21438	21442	21449
	21453	21458	21464	21467	21471	21482	21512	21517	21531	21535	21539	21542
	21546	21549	21584	21603	21615	21618	21650	21655	21664	21697	21705	21715
	21730	21771	21786	21794	21808	21858	21888	21903	21951	21972	21978	21994
	21997	22002	22008	22011	22015	22020	22054	22057	22092	22424	22446	22457
	22308	22531	22541	22567	22581	22603	22609	22623	22627	22666	22673	22704
	22715	22721	22745	22761	22764	22772	22785	22939	22955	22985	23007	23017

B	1	MICRO2	1M(01)	J	5	MICRO2	1M(01)	E	10	MICRO2	1M(01)
C	1	MICRO2	1M(01)	K	5	MICRO2	1M(01)	F	10	MICRO2	1M(01)
D	1	MICRO2	1M(01)	L	5	MICRO2	1M(01)	G	10	MICRO2	1M(01)
E	1	MICRO2	1M(01)	M	5	MICRO2	1M(01)	H	10	MICRO2	1M(01)
F	1	MICRO2	1M(01)	N	5	MICRO2	1M(01)	I	10	MICRO2	1M(01)
G	1	MICRO2	1M(01)	B	6	MICRO2	1M(01)	J	10	MICRO2	1M(01)
H	1	MICRO2	1M(01)	C	6	MICRO2	1M(01)	K	10	MICRO2	1M(01)
I	1	MICRO2	1M(01)	D	6	MICRO2	1M(01)	L	10	MICRO2	1M(01)
J	1	MICRO2	1M(01)	E	6	MICRO2	1M(01)	M	10	MICRO2	1M(01)
K	1	MICRO2	1M(01)	F	6	MICRO2	1M(01)	N	10	MICRO2	1M(01)
L	1	MICRO2	1M(01)	G	6	MICRO2	1M(01)	B	11	MICRO2	1M(01)
M	1	MICRO2	1M(01)	H	6	MICRO2	1M(01)	C	11	MICRO2	1M(01)
N	1	MICRO2	1M(01)	I	6	MICRO2	1M(01)	D	11	MICRO2	1M(01)
B	2	MICRO2	1M(01)	J	6	MICRO2	1M(01)	E	11	MICRO2	1M(01)
C	2	MICRO2	1M(01)	K	6	MICRO2	1M(01)	F	11	MICRO2	1M(01)
D	2	MICRO2	1M(01)	L	6	MICRO2	1M(01)	G	11	MICRO2	1M(01)
E	2	MICRO2	1M(01)	M	6	MICRO2	1M(01)	H	11	MICRO2	1M(01)
F	2	MICRO2	1M(01)	N	6	MICRO2	1M(01)	I	11	MICRO2	1M(01)
G	2	MICRO2	1M(01)	B	7	MICRO2	1M(01)	J	11	MICRO2	1M(01)
H	2	MICRO2	1M(01)	C	7	MICRO2	1M(01)	K	11	MICRO2	1M(01)
I	2	MICRO2	1M(01)	D	7	MICRO2	1M(01)	L	11	MICRO2	1M(01)
J	2	MICRO2	1M(01)	E	7	MICRO2	1M(01)	M	11	MICRO2	1M(01)
K	2	MICRO2	1M(01)	F	7	MICRO2	1M(01)	N	11	MICRO2	1M(01)
L	2	MICRO2	1M(01)	G	7	MICRO2	1M(01)	B	12	MICRO2	1M(01)
M	2	MICRO2	1M(01)	H	7	MICRO2	1M(01)	C	12	MICRO2	1M(01)
N	2	MICRO2	1M(01)	I	7	MICRO2	1M(01)	D	12	MICRO2	1M(01)
B	3	MICRO2	1M(01)	J	7	MICRO2	1M(01)	E	12	MICRO2	1M(01)
C	3	MICRO2	1M(01)	K	7	MICRO2	1M(01)	F	12	MICRO2	1M(01)
D	3	MICRO2	1M(01)	L	7	MICRO2	1M(01)	G	12	MICRO2	1M(01)
E	3	MICRO2	1M(01)	M	7	MICRO2	1M(01)	H	12	MICRO2	1M(01)
F	3	MICRO2	1M(01)	N	7	MICRO2	1M(01)	I	12	MICRO2	1M(01)
G	3	MICRO2	1M(01)	B	8	MICRO2	1M(01)	J	12	MICRO2	1M(01)
H	3	MICRO2	1M(01)	C	8	MICRO2	1M(01)	K	12	MICRO2	1M(01)
I	3	MICRO2	1M(01)	D	8	MICRO2	1M(01)	L	12	MICRO2	1M(01)
J	3	MICRO2	1M(01)	E	8	MICRO2	1M(01)	M	12	MICRO2	1M(01)
K	3	MICRO2	1M(01)	F	8	MICRO2	1M(01)	N	12	MICRO2	1M(01)
L	3	MICRO2	1M(01)	G	8	MICRO2	1M(01)	B	13	MICRO2	1M(01)
M	3	MICRO2	1M(01)	H	8	MICRO2	1M(01)	C	13	MICRO2	1M(01)
N	3	MICRO2	1M(01)	I	8	MICRO2	1M(01)	D	13	MICRO2	1M(01)
B	4	MICRO2	1M(01)	J	8	MICRO2	1M(01)	E	13	MICRO2	1M(01)
C	4	MICRO2	1M(01)	K	8	MICRO2	1M(01)	F	13	MICRO2	1M(01)
D	4	MICRO2	1M(01)	L	8	MICRO2	1M(01)	G	13	MICRO2	1M(01)
E	4	MICRO2	1M(01)	M	8	MICRO2	1M(01)	H	13	MICRO2	1M(01)
F	4	MICRO2	1M(01)	N	8	MICRO2	1M(01)	I	13	MICRO2	1M(01)
G	4	MICRO2	1M(01)	B	9	MICRO2	1M(01)	J	13	MICRO2	1M(01)
H	4	MICRO2	1M(01)	C	9	MICRO2	1M(01)	K	13	MICRO2	1M(01)
I	4	MICRO2	1M(01)	D	9	MICRO2	1M(01)	L	13	MICRO2	1M(01)
J	4	MICRO2	1M(01)	E	9	MICRO2	1M(01)	M	13	MICRO2	1M(01)
K	4	MICRO2	1M(01)	F	9	MICRO2	1M(01)	N	13	MICRO2	1M(01)
L	4	MICRO2	1M(01)	G	9	MICRO2	1M(01)	B	14	MICRO2	1M(01)
M	4	MICRO2	1M(01)	H	9	MICRO2	1M(01)	C	14	MICRO2	1M(01)
N	4	MICRO2	1M(01)	I	9	MICRO2	1M(01)	D	14	MICRO2	1M(01)
B	5	MICRO2	1M(01)	J	9	MICRO2	1M(01)	E	14	MICRO2	1M(01)
C	5	MICRO2	1M(01)	K	9	MICRO2	1M(01)	F	14	MICRO2	1M(01)
D	5	MICRO2	1M(01)	L	9	MICRO2	1M(01)	G	14	MICRO2	1M(01)
E	5	MICRO2	1M(01)	M	9	MICRO2	1M(01)	H	14	MICRO2	1M(01)
F	5	MICRO2	1M(01)	N	9	MICRO2	1M(01)	I	14	MICRO2	1M(01)
G	5	MICRO2	1M(01)	B	10	MICRO2	1M(01)	J	14	MICRO2	1M(01)
H	5	MICRO2	1M(01)	C	10	MICRO2	1M(01)	K	14	MICRO2	1M(01)
I	5	MICRO2	1M(01)	D	10	MICRO2	1M(01)	L	14	MICRO2	1M(01)

M	14	MICRO2	1M(01)
N	14	MICRO2	1M(01)
B	15	MICRO2	1M(01)
C	15	MICRO2	1M(01)
D	15	MICRO2	1M(01)
E	15	MICRO2	1M(01)
F	15	MICRO2	1M(01)
G	15	MICRO2	1M(01)
H	15	MICRO2	1M(01)
I	15	MICRO2	1M(01)
J	15	MICRO2	1M(01)
K	15	MICRO2	1M(01)
L	15	MICRO2	1M(01)
M	15	MICRO2	1M(01)
N	15	MICRO2	1M(01)
B	16	MICRO2	1M(01)
C	16	MICRO2	1M(01)
D	16	MICRO2	1M(01)
E	16	MICRO2	1M(01)
F	16	MICRO2	1M(01)
G	16	MICRO2	1M(01)
H	16	MICRO2	1M(01)
I	16	MICRO2	1M(01)
J	16	MICRO2	1M(01)
K	16	MICRO2	1M(01)
L	16	MICRO2	1M(01)

Cross Reference Listing - Field Names and Defined Values

23021	23030	23034	23038	23049	23090	23095	23106	23154	23158	23167	23171
23175	23186	23204	23210	23214	23225	23230	23234	23258	23306	23310	23332
23342	23357	23464	23479	23493	23500	23569	23573	23669	23674	23693	23760
23782	23807	23816	23844	23848	23875	23897	23969	23974	24010	24014	24024
24078	24088	24102	24112	24245	24250	24327	24333	24433	24442	24489	24500
24506	24515	24551	24560	24575	24629	24635	24640	24647	24655	24664	24669
24680	24683	24688	24697	24707	24715	24721	24796	24801	24834	24843	24859
24863	24883	24925	24929	24933	24948	24952	24959	24963	24969	24997	25025
25045	25090	25102	25165	25177	25182	25203	25243	25283	25292	25394	25401
25611	25625	25629	25655	25676	25680	25686	25800	25817	25821	25827	25832
25837	25842	26095	26123	26125	26135	26153	26168	26176	26184	26211	26216
26348	26435	26444	26472	26475	26480	26483	26486	26489	26496	26603	26615
26620	26623	26626	26789	26793	26847	26852	27007	27029	27032	27035	27038
27292	27315	27325	27328	27331	27374	27378	27427	27498	27502	27512	27525
27533	27668	27682	27692	27696	27737	27796	27820	27989	28014	28060	28154
28171	28240	28245	28284	28327	28333	28337	28476	28499	28506	28532	28537
28554	28562	28593	28597	28601	28606	28617	28622	28632	28636	28640	28660
28665	28669	28764	28773	28783	28792	28796	28809	28823	28827	28831	28835
29112	29115	29124	29133	29137	29141	29166	29176	29218	29249	29268	29360
29402	29409	29488	29511	29534	29538	29542	29559	29578	29584	29589	29594
29599	29604	29615	29659	29663	29669	29674	29679	29684	29697	29710	29715
29721	29726	29754	29758	29768	29821	29828	29831	29842	29849	29868	29876
29902	29911	29949	29981	29986	30001	30005	30009	30011	30033	30049	30089
30094	30125	30130	30180	30185	30225	30242	30247	30252	30256	30265	30271
30284	30289	30331	30342	30346	30369	30433	30443	30451	30535	30584	30599
30606	30624	30630	30635	30640	30648	30713	30755	30759	30785	30832	30836
30848	30852	30856	30860	30864	30873	30878	30887	30908	30926	30967	31012
31048	31061	31119	31123	31126	31133	31139	31163	31181	31227	31230	31241
31312	31333	31348	31426	31469	31479	31522	31548	31574	31587	31649	31688
31703	31707	31714	31911	31941	32044	32049	32093	32099	32131	32135	32172
32176	32384	32395	32399	32403	32482	32511	32514	32525	32573	32628	32634
32638	32644	32648	32654	32658	32663	32667	32670	32686	32701	32706	32719
32723	32727	32730	32733	32737	32813	32822	32828	32832	32836	32843	32847
32857	32861	32878	32883	32903	32915	32919	32925	32929	32937	32941	32947
32952	32983	32991	32996	33011	33018	33026	33030	33037	33040	33046	33051
33082	33119	33155	33500	33514	33518	33522	33525	33541	33545	33549	33553
33575	33609	33614	33637	33648	33668	33672	33676	33720	33726	33737	33782
33809	33902	33916	33956	33998	34003	34020	34028	34033	34036	34071	34075
34088	34092	34132	34145	34149	34153	34157	34167	34171	34177	34185	34197
34240	34259	34263	34267	34273	34279	34285	34290	34294	34297	34301	34318
34372	34393	34405	34408	34435	34440	34458	34462	34494	34498	34501	34542
34553	34560	34569	34594	34599	34610	34623	34658	34679	34701	34707	34713
34724	34728	34732	34837	34841	34845	34849	34863	34886	34890	34899	34903
34952	34967	34982	35030	35042	35046	35058	35070	35074	35086	35139	35147
35151	35159	35163	35172	35176	35179	35190	35195	35199	35208	35217	35221
35230	35234	35243	35251	35260	35268	35271	35279	35287	35303	35314	35356
35359	35368	35372	35375	35390	35393	35417	35422	35455	35464	35468	35473
35481	35520	35539	35546	35557	35566	35616	35635	35639	35746	35759	35763
35774	35786	35790	35794	35805	35829	35845	35868	35876	35880	35888	35900
35943	35982	35986	36004	36018	36038	36046	36051	36060	36063	36065	36125
36136	36160	36166	36254	36263	36271	36275	36279	36283	36303	36315	36327
36331	36353	36360	36469	36474	36478	36491	36495	36527	36553	36558	36562
36575	36579	36630	36683	36700	36705	36715	36718	36729	36732	36736	36739
36745	36755	36763	36767	36771	36776	36781	36793	36799	36802	36808	36812
36841	36853	36871	36875	36882	36885	36904	36908	36914	36916	36922	37098

Cross Reference Listing - Field Names and Defined Values

	37114	37119	37129	37135	37194	37204	37226	37255	37270	37274	37278	37282
	37294	37298	37302	37306	37311	37316	37321	37326	37344	37360	37368	37379
	37384	37388	37419	37431	37436	37440	37486	37651	37657	37666	37678	37693
	37697	37704	37718	37722	37736	37743	37747	37755	37768	37776	37787	37819
	37830	37844	37867	37872	37876	37908	37911	38005	38091	38105	38112	38152
	38163	38173	38213	38222	38225	38228	38262	38278	38292	38296	38301	38354
	38357	38370	38379	38389	38399	38404	38408	38433	38443	38466	38472	38476
	38482	38486	38492	38561	38570	38577	38653	38662	38698	38705	38711	38716
	38721	38728	38756	38766	38806	38850	38855	38916	38927	39011	39038	39066
	39070	39078	39090	39111	39157	39209	39215	39268	39274	39338	39354	39364
	39385	39403	39407	39411	39415	39419	39430	39462	39474	39644	39692	39722
	39725	39736	39739	39759	39801	39805	39809	39813	39833	39874	39908	39918
	39948	39959	39964	39974	39996	40033	40039	40054	40140	40149	40153	40179
	40217	40225	40232	40237	40241	40297	40309	40313	40324	40327	40375	40380
	40390	40398	40515	40566	40576	40584	40593	40597	40605	40692	40701	40705
	40755	40762	40766	40814	40821	40854	40862	40873	40901	40929	41024	41030
	41035	41042	41062	41068	41073	41080	41100	41111	41118	41138	41143	41148
	41154	41257	41268	41279	41332	41346	41366	41423	41451	41647	41651	41668
	41672	41813	41863	41895	41926	42014	42019	42044	42052	42056	42060	42113
	42175	42252	42285	42318	42379	42388	42393	42447	42486	42492	42541	42550
	42565	42570	42619	42629	42661	42666	42673	42694	42739	42788	42808	42820
	42886	42938	42966	43022	43027	43079	43103	43108	43181	43190	43213	44361
	44435	44442	44522	44533	44550	44708	44713	44750	44798	44895	44902	44937
	44944	45001	45016	45025	45056	45066	45274	45336	45368	45451	45455	45533
	45539	45547	45553	45598	45673	45780	45785	45908	45913	45982	46076	46084
	46179	46472	46566	46771	46775	46869	46874	46878	46951	47090	47097	47108
	47161	47172	47207	47218	47247	47258	47281	47298	47336	47352	47369	47573
	47646	47651	47654	47740	47749	47757	47780	47808	47828	47832	47835	47935
	47942	47982										
R.Q	3029 #	5573	5578	5586	5595	5646	7733	7737	8100	8291	8362	8373
	8420	8492	8506	8909	8926	9018	9022	9027	9314	9323	9381	9405
	9409	9435	9440	9445	9501	10679	10732	10804	10991	10995	11001	11145
	11157	11233	11243	11334	11440	11570	11629	11756	11977	12046	12160	12232
	12833	13052	13089	13156	13232	13453	13618	13766	13785	14082	14309	14328
	14493	14639	14940	15238	15299	15304	16440	16447	16505	16986	18363	18367
	18371	18651	18671	18678	19173	19177	19199	19203	19207	19211	19215	19219
	19302	19352	19357	19460	19464	19473	19492	20458	21005	21030	21281	21434
	21571	21600	21624	21678	21693	21719	21726	21801	21896	22502	22561	22730
	22983	24317	24371	24546	24565	24572	24847	24988	26255	26283	27540	28038
	28159	28717	28729	28737	29415	29524	33115	33487	34964	34974	35098	35102
	35130	35143	35239	35247	35265	35342	35398	35607	35629	35655	36687	36696
	36865	39222	40212	40221	40321	41174	41184	41775	41785	41824	41973	41979
	41984	41995	42097	42102	42489	42611	46295	46398	46423	46448	46495	47359
	47381											
R.S	3030 #	5438	5448	5526	5546	5552	5568	5667	5726	5754	5880	5887
	5892	5903	5907	5924	6260	6277	6436	6656	6678	6701	6743	6746
	6760	6806	6811	6822	6864	6879	6886	6951	6956	6969	6987	7027
	7192	7215	7238	7287	7392	7396	7423	7447	7469	7492	7514	7536
	7655	7667	7725	7804	7823	8097	8103	8127	8131	8135	8139	8150
	8153	8455	8467	8992	8995	9309	9498	9825	11807	12881	13073	13571
	13576	13581	13595	13609	13629	13866	14093	14102	14762	14789	14794	14984
	15113	15191	15471	15736	15770	15799	15850	15916	16203	16211	16225	16234
	16238	16242	16255	16259	16262	16267	16284	16414	16425	16428	16456	16510
	16548	16567	16593	16722	16733	16761	16791	16797	16802	16833	16844	16926
	16942	16994	17030	17052	17195	17199	17210	17214	17218	17221	17242	17303

Cross Reference Listing - Field Names and Defined Values

17486	17491	17550	17578	17587	17616	17619	17629	18202	18378	18382	18685
18688	19394	19455	19575	19639	19722	19733	19738	20110	20116	20186	20195
20224	20229	20239	20248	20257	20265	20274	20283	20292	20300	20309	20318
20327	20335	20344	20418	20433	20441	20444	20479	20503	20508	20513	20518
20530	21099	21147	21152	21158	21201	21232	21253	21267	21272	21819	21823
22642	22751	22949	22991	23045	23056	23073	23085	23102	23110	23115	23119
23137	23144	23182	23190	23196	23201	23248	23253	23261	23265	23269	23288
23295	23319	23324	23329	23483	23520	23526	23530	23535	23540	23544	23550
23555	23684	23702	23725	23821	23825	23829	23833	23838	23880	23884	23888
23936	23992	24001	24027	24115	24189	24215	24219	24238	24307	24323	24376
24437	24531	24569	24644	24701	24712	24735	24757	24778	24789	24808	24819
24828	24854	24873	24992	25015	25056	25080	25113	25122	25128	25207	25262
25302	25307	25322	25357	25363	25371	25376	25379	25388	25405	25639	25643
25691	26130	26150	26189	26198	26223	26237	26240	26274	26277	26306	26333
26509	26515	26636	26641	26670	26674	26678	26682	27097	27112	27117	27121
27125	27129	27136	27145	27177	27182	27186	27258	27264	27271	27275	27303
27336	27341	27356	27360	27423	27441	27465	27677	27723	27747	27787	27791
27813	27919	27949	27959	27970	27992	28017	28021	28030	28078	28086	28109
28121	28125	28128	28131	28134	28208	28217	28278	28302	28306	28347	28401
28411	28942	28945	28962	28971	28978	28982	29083	29097	29106	29159	29201
29238	29272	29365	29376	29412	29462	29469	29481	29623	29627	29649	29654
29692	29702	29706	29748	29776	29814	29834	29859	29872	29881	29885	29891
29895	29924	30025	30039	30045	30210	30229	30359	30363	30400	30488	30505
30521	30567	30708	30741	30768	30826	31039	31043	31068	31128	31172	31175
31178	31218	31224	31233	31235	31248	31252	31256	31260	31285	31290	31299
31305	31460	31482	31493	31719	31775	31860	31887	31914	31932	31947	32029
32054	32058	32071	32089	32096	32501	32535	32554	33131	33134	33164	33167
33175	33492	33509	33642	33663	33776	33788	33792	33798	33813	33858	33862
33898	33907	33948	33952	33959	34067	34084	34161	34192	34253	34304	34314
34349	34354	34358	34368	34432	34446	34450	34489	34509	34535	34607	34613
34666	34673	34685	34688	34692	34696	34754	34758	34867	34872	34876	34956
35135	35353	35430	35612	35624	35644	35770	35884	35892	35957	35991	35996
36012	36026	36076	36131	36173	36267	36293	36341	36348	36357	36517	36522
36791	36817	36857	36861	36878	36891	36899	36911	36919	36929	36935	36941
36965	37199	37230	37235	37240	37265	37287	37330	37356	37364	37733	37773
37810	37816	37980	37984	37989	37993	37997	38001	38275	38373	38456	38460
38566	38658	38775	38832	38844	38970	38974	39002	39006	39053	39060	39096
39103	39117	39162	39166	39184	39329	39335	39341	39381	39434	39441	39445
39449	39452	39478	39493	39507	39729	39743	39816	39889	39895	39899	39902
39935	39938	39979	39983	39987	39990	40049	40059	40065	40070	40095	40184
40199	40207	40249	40260	40265	40300	40365	40393	40439	40441	40444	40482
40487	40492	40506	40531	40537	40555	40621	40630	40636	40655	40713	40718
40722	40774	40778	40781	40869	40905	40909	40913	40933	40937	40941	40994
41021	41059	41097	41135	41221	41247	41301	41457	41460	41587	41742	41952
42076	42191	42199	42375	42383	42402	42411	42426	42434	42442	42477	42530
42537	42545	42557	42579	42603	42643	42652	42715	42743	42776	42782	42802
42829	42845	42891	42925	42931	42952	42975	42991	43032	43066	43072	43091
43130	43166	44367	44424	44454	44472	44483	44500	44511	44527	44544	44578
44607	44623	44640	44657	44668	44686	44697	44717	44729	44746	44757	44774
44785	44835	44889	44913	45010	45031	45038	45061	45073	45092	45106	45114
45125	45136	45147	45172	45185	45210	45225	45245	45290	45373	45442	45460
45479	45523	45564	45589	45593	45607	45625	45640	45648	45657	45702	45706
45717	45735	45750	45758	45772	45800	45832	45919	45942	45977	46023	46042
46170	46175	46212	46230	46364	46372	46377	46382	46521	46560	46563	46598
46632	46665	46980	46983	47052	47059	47204	47244	47326	47331	47341	47396

Cross Reference Listing - Field Names and Defined Values

	47958	47973	48006	48012	48048							
XM.Q	3021 #	8296	8300	8311	8315	27727	28849	28865	29227	29369	32037	41341
XM.R	3020 #	8087	8332	8387	9147	9152	9157	9162	9305	9706	14649	14947
	15406	18198	18217	18274	18286	18315	18330	18914	19096	19187	19248	19283
	19329	19381	19444	19564	19571	19578	20207	20219	20233	22496	22556	22596
	22661	22710	23449	23453	23766	23978	23988	24039	24082	24128	24195	24225
	24231	24256	24262	24313	24342	26302	29129	29145	29162	29275	29338	29349
	29554	29639	30579	32111	32118	32442	32463	32498	33992	38167	41206	41225
	41305	41801	41805	42107	42356	42438	42655	42719	42767	42792	42797	42916
	42942	42947	43057	43083	43087	43197	43202	43207	43225	43230	43235	44372
	44377	44391	44396	44488	44492	44504	44507	44583	44588	44600	44603	44673
	44678	44690	44693	44762	44766	44778	44781	44840	44845	44859	44864	44949
	44954	44959	44968	44972	45078	45083	45088	45096	45101	45192	45199	45215
	45220	45295	45300	45313	45318	45378	45383	45397	45402	45465	45470	45484
	45489	45611	45616	45630	45635	45721	45726	45740	45745	45817	45837	45842
	45856	45861	45925	45931	45947	45953	46028	46033	46047	46052	46882	46889
	46989	46999	47008	47018								
XM.S	3022 #	5467	7031	7198	8146	8320	8897	8903	9269	9274	9279	9285
	9290	9295	11319	11445	11561	15985	17070	17454	17499	17625	17638	18970
	18975	18980	18985	18990	18995	19000	19004	19524	19658	19663	19668	19673
	19678	19683	19688	19692	20144	20168	20764	21106	21126	21131	21243	21248
	22418	22514	22525	22574	22615	22652	22697	23730	23734	23786	24446	24456
	24752	24768	24792	24824	24876	24936	25035	25069	25084	25098	25213	25312
	25318	25325	25382	25601	26088	26099	26417	26423	26439	27752	27979	28003
	28686	28691	29154	29184	29234	29257	29284	29421	29427	29565	29570	29574
	29808	29854	30083	30099	30174	30189	30205	30220	30324	30336	30427	30438
	30484	30644	30664	30668	30673	30723	30896	30902	30936	30941	30946	30982
	30990	31032	31411	31421	31430	31435	31891	32151	32158	33146	33171	33465
	33473	33571	33623	34619	34669	36295	36465	36549	41251	41276	41591	41603
	41619	41947	42081	47015								
Z.S	3028 #	6170	6269	6814	6843	7584	13460	16654	16849	27062	27067	31673
	32371	32379	32790	35896								
NEXT	3033 #	5485	5496	5508	5651	5676	5702	5737	5818	5882	5888	5974
	5978	6001	6459	7310	7557	7569	7618	7702	7710	7805	7824	7829
	7833	7864	7868	7877	7903	7949	8089	8154	8168	8240	8245	8261
	8266	8278	8301	8316	8321	8351	8357	8363	8368	8373	8378	8416
	8420	8436	8441	8460	8472	8501	8511	8522	8536	8553	8564	8590
	8596	8606	8620	8636	8653	8679	8685	8695	8709	8727	8737	8763
	8770	8780	8807	8814	8824	8851	8858	8869	8942	8946	8950	8992
	8996	9072	9199	9203	9238	9242	9319	9336	9519	9527	9603	9614
	9619	9637	9648	9653	9687	9721	9731	9765	9782	9835	10694	10699
	10704	10822	10831	10868	10875	10879	10926	10930	10934	10992	10996	11002
	11016	11026	11029	11034	11037	11136	11149	11154	11163	11170	11174	11181
	11380	11493	11499	11699	11706	11712	11715	11777	11790	11899	11910	11930
	11935	12137	12142	12146	12162	12229	12233	12360	12365	12372	12542	12576
	12612	12628	12646	12700	12716	12721	12734	12738	12745	12892	12959	12964
	12969	12979	12985	12990	13320	13343	13348	13573	13596	13641	13665	13678
	13693	13712	13716	13867	13974	13987	13992	14098	14113	14259	14513	14520
	14795	14804	14816	15120	15147	15198	15232	15239	15243	15343	15407	15418
	15422	15426	15551	15837	15842	16015	16047	16053	16060	16064	16074	16294
	16544	16742	16755	16782	16792	16852	16883	16894	16950	17076	17102	17118
	17123	17517	17547	17557	17566	18241	18245	18279	18291	18321	18685	18688
	18711	18733	18765	18769	18772	18910	19102	19105	19109	19113	19117	19121
	19125	19129	19150	19193	19196	19200	19204	19208	19212	19216	19220	19325
	19421	19424	19427	19489	19590	19595	19728	20419	20487	20504	20509	20514

Cross Reference Listing - Field Names and Defined Values

```
BO.IRD1_SUB                5477    5482 #
BO.POWER_UP                5321 #
BO.R-B_WARM_CHECK          5389    5406 #
BO.R-H_WARM_CHECK          5393    5396 #
BO.READ_RESTART_ROUTINE    5772    5787 #
BO.READ_RPB_HEADER         5728 #  5824
BO.READ_SUB                5564    5569    5579    5596    5684 #
BO.RESTART                 5399    5409    5415 #
BO.RESTART_HALT            5373 #  6248
BO.RESTART_SEARCH1         5556    5696 #
BO.RESTART_SEARCH2         5699    5710 #
BO.START_SEARCH            5528 #  5716
BO.TEST_ACLO               5337 #  5346
BO.TRANSFER_ROMS           5669 #  5682
BO.WRITE_UBA_MAP           5622    5628 #
BO.WRITE_WALK0             5583    5593 #
BO.WRITE_WALK1             5560    5576 #
BO.WRITE_ZERO              5553    5566 #
CH.CHME                    36230 # 36369   36369   36369   36369   36369   36369
CH.CHMK                    36225 # 36376   36376   36376   36376   36376   36376
CH.CHMS                    36235 # 36383   36383   36383   36383   36383   36383
CH.CHMU                    36239 # 36390   36390   36390   36390   36390   36390
CH.CHMX.10                 36233   36238   36242   36246 #
CH.CHMX.20                 36228   36261 #
CH.CHMX.30                 36259   36265 #
CH.CHMX.40                 36272   36276   36280   36284   36291 #
CH.CHMX.45                 36316   36322 #
CH.CHMX.50                 36332   36339 #
CI.PATCH.RET               21678   21695 #
CM-SET.CLEAR               42501 # 43394   43394   43395   43395   43397   43397   43398   43398
CM-TEST.OSR                38616   38622   42428   42645   42692 #
CM-TEST.OSR.CONT           42707   42709 #
CM-TEST.OSR1               42695   42701 #
CM.ADC-MEM                 41497 # 43273   43273   43276   43276
CM.ADC-REG                 41478 # 43272   43272   43275   43275
CM.ADD-MEM                 41850 # 43279   43279   43282   43282
CM.ADD-REG                 41856 # 43278   43278   43281   43281
CM.ASH                     41295 # 43284   43284   43285   43285   43287   43287   43288   43288
CM.ASH-CHECKC              41357 # 41461
CM.ASH-CHECKV              41316   41339 #
CM.ASH-CHECKV.-1           41308 # 41323
CM.ASH-CHECKV.1            41342   41349 #
CM.ASH-NEG                 41329   41369 #
CM.ASH-SETC                41364 # 41385
CM.ASH-STNEG               41325 # 41337
CM.ASH-TESTC               41375 # 41465
CM.ASH.1                   41306   41319 #
CM.ASHC                    41398 # 43290   43290   43291   43291   43293   43293   43294   43294
CM.ASHC-NEG                41420   41445 #
CM.ASHC-POS                41414   41431 #
CM.ASHC-STNEG              41416 # 41429
CM.ASHC-TESTC              41455 #
CM.ASHC-TESTV              41435   41441 #
CM.ASL-MEM                 41274 # 43297   43297   43300   43300
CM.ASL-REG                 41245 # 43296   43296   43299   43299
```

Cross Reference Listing - Field Names and Defined Values

```
CM.ASL-SHIFT      41255 #
CM.ASR-MEM        41204 # 43303   43303   43306   43306
CM.ASR-REG        41219 # 43302   43302   43305   43305
CM.BIC-MEM        42305 # 43328   43328   43331   43331   43334   43334   43337   43337
CM.BIC-REG        42311 # 43327   43327   43330   43330   43333   43333   43336   43336
CM.BIS-MEM        42239 # 43340   43340   43343   43343   43346   43346   43349   43349
CM.BIS-REG        42245 # 43339   43339   43342   43342   43345   43345   43348   43348
CM.BIT-MEM        42272 # 43352   43352   43355   43355   43358   43358   43361   43361
CM.BIT-REG        42278 # 43351   43351   43354   43354   43357   43357   43360   43360
CM.BR             42333 # 43385   43385   43386   43386
CM.BRCND          42344 # 43308   43308   43309   43309   43311   43311   43312   43312   43314   43314   43315
                  43315   43317   43317   43318   43318   43321   43321   43322   43322   43324   43324   43325
                  43325   43363   43363   43364   43364   43366   43366   43367   43367   43369   43369   43370
                  43370   43372   43372   43373   43373   43376   43376   43377   43377   43379   43379   43380
                  43380   43388   43388   43389   43389   43391   43391   43392   43392
CM.BRCND-BRANCH   42336   42354 #
CM.CLR-MEM        41003 # 43401   43401   43404   43404
CM.CLR-REG        40984 # 43400   43400   43403   43403
CM.CMP-MEM        41913 # 43407   43407   43410   43410   43413   43413   43416   43416
CM.CMP-REG        41919 # 43406   43406   43409   43409   43412   43412   43415   43415
CM.COM-MEM        41189 # 43419   43419   43422   43422
CM.COM-REG        41168 # 43418   43418   43421   43421
CM.DEC-MEM        41040 # 43425   43425   43428   43428
CM.DEC-REG        41019 # 43424   43424   43427   43427
CM.DIV            42074 # 43431   43431   43432   43432   43434   43434   43435   43435
CM.DIV-CALL.SUB   42091   42105 #
CM.DIV-CLEAR.C    42129   42138   42159   42164   42169 #
CM.DIV-END        42150   42156 #
CM.DIV-FIX.REM    42095 #
CM.DIV-TEST.OV    42103   42111 #
CM.DIV-TEST.OV1   42099   42131 #
CM.DIV-TEST.OV2   42114   42122 #
CM.DIV-TEST.REM   42141 # 42154
CM.DIV-TEST.REM1  42119   42125   42134   42145 #
CM.INC-MEM        41078 # 43450   43450   43453   43453
CM.INC-REG        41057 # 43449   43449   43452   43452
CM.IRDX           42860   43258 #
CM.JMP-MEM        42462 # 43459   43459   43462   43462   43465   43465   43468   43468
CM.JMP-REG        42458 # 43458   43458   43461   43461   43464   43464   43467   43467
CM.JSR            42366 # 43470   43470   43471   43471   43473   43473   43474   43474   43476   43476   43477
                  43477   43479   43479   43480   43480   43482   43482   43483   43483   43486   43486   43487
                  43487   43489   43489   43490   43490   43492   43492   43493   43493
CM.JSR-MODE.OK    42373 # 42404
CM.JSR-TEST.MODE  42369   42409 #
CM.JSR-TEST.SP    42400 # 42412
CM.JSR-WRITE.PC   42380   42391 #
CM.JSR-WRITE.PC.1 42389   42395 #
CM.JSR-WRITE.REG  42376   42384   42386 #
CM.MFP-PC         42542   42548 #
CM.MFPD-CHECK.SP  42559   42568 #
CM.MFPD-MEM       42554 # 43496   43496
CM.MFPD-REG       42528 # 43495   43495
CM.MFPD-TEST.MODE 42535 # 42581
CM.MFPD-TEST.SP   42531   42577 #
CM.MFPI-MEM       42555 # 43499   43499
```

Cross Reference Listing - Field Names and Defined Values

```
CM.MFPI-REG              42527 # 43498   43498
CM.MOV-MEM               41833 # 43508   43508   43511   43511   43514   43514   43517   43517
CM.MOV-REG               41811 # 43507   43507   43510   43510
CM.MOV-WRITE             41802   41817 #
CM.MOVB-REG              41799 # 43513   43513   43516   43516
CM.MTP-WRITE             42612   42617 #
CM.MTPD                  42594 # 43501   43501   43502   43502
CM.MTPI                  42595 # 43504   43504   43505   43505
CM.MTPI-TEST.REGMOD      42604   42609 #
CM.MUL                   41945 # 43519   43519   43520   43520   43522   43522   43523   43523
CM.MUL-NEG               41964   41967 #
CM.MUL-NEG.WRT           41986   41988 #
CM.MUL-NEG.WRT1          42003   42009   42012 #
CM.MUL-NEG.WRT2          41970   41977 # 41991
CM.MUL-NEG.WRT3          41974   41981   41996   42000 #
CM.MUL-POS               41959   42024 #
CM.MUL-POS.WRT2          42029   42035 #
CM.MUL-READ.PSW          42021   42039 #
CM.NEG-MEM               41116 # 43526   43526   43529   43529
CM.NEG-REG               41095 # 43525   43525   43528   43528
CM.NOP                   40996   41108   41265   42351 # 42357   42493
CM.OS.MOD                42879 # 43273   43273   43276   43276   43279   43279   43282   43282   43297   43297   43300
                         43300   43303   43303   43306   43306   43328   43328   43331   43331   43334   43334   43337
                         43337   43340   43340   43343   43343   43346   43346   43349   43349   43419   43419   43422
                         43422   43425   43425   43428   43428   43450   43450   43453   43453   43526   43526   43529
                         43529   43975   43975   43978   43978   43982   43982   43985   43985   43997   43997   44000
                         44000   44009   44009   44012   44012   44015   44015   44018   44018   44021   44021   44024
                         44024   44048   44048   44049   44049   44051   44051   44052   44052
CM.OS.MOD-ADEC.PC        42939   42998 #
CM.OS.MOD-AUTO.DEC       42922   43003 #
CM.OS.MOD-ODD.CHECK1     42893   42897   42918   42928   42944   42949   42963 # 43007
CM.OS.MOD-ODD.CHECK2     42902   42972 #
CM.OS.MOD-ODD.CHECK2A    42977   42982 #
CM.OS.MOD-ODD.CHECK3     42907   42988 #
CM.OS.RED                42733 # 43278   43278   43279   43279   43281   43281   43282   43282   43284   43284   43285
                         43285   43287   43287   43288   43288   43290   43290   43291   43291   43293   43293   43294
                         43294   43327   43327   43328   43328   43330   43330   43331   43331   43333   43333   43334
                         43334   43336   43336   43337   43337   43339   43339   43340   43340   43342   43342   43343
                         43343   43345   43345   43346   43346   43348   43348   43349   43349   43351   43351   43352
                         43352   43352   43352   43354   43354   43355   43355   43355   43355   43357   43357   43358
                         43358   43358   43358   43360   43360   43361   43361   43361   43361   43406   43406   43407
                         43407   43407   43407   43409   43409   43410   43410   43410   43410   43412   43412   43413
                         43413   43413   43413   43415   43415   43416   43416   43416   43416   43431   43431   43432
                         43432   43434   43434   43435   43435   43496   43496   43499   43499   43507   43507   43508
                         43508   43510   43510   43511   43511   43513   43513   43514   43514   43516   43516   43517
                         43517   43519   43519   43520   43520   43522   43522   43523   43523   44008   44008   44009
                         44009   44011   44011   44012   44012   44043   44043   44046   44046
CM.OS.RED-ADEC.PC        42789   42852 #
CM.OS.RED-AUTO.DEC       42774   42862 #
CM.OS.RED-IMED           42806 # 42959
CM.OS.RED-IMMEDIATE      42809   42857 #
CM.OS.RED-ODD.CHECK1     42745   42749   42769   42779   42794   42799   42817 # 42855   42866   43001
CM.OS.RED-ODD.CHECK2     42754   42826 #
CM.OS.RED-ODD.CHECK2A    42831   42836 #
CM.OS.RED-ODD.CHECK3     42759   42842 #
```

Cross Reference Listing - Field Names and Defined Values

```
CM.OS.RED-REGPC          42740   42812 # 42888
CM.OS.WRT                43020 # 43401   43401   43404   43404   43459   43459   43462   43462   43465   43465   43468
                         43468   43470   43470   43471   43471   43473   43473   43474   43474   43476   43476   43477
                         43477   43479   43479   43480   43480   43482   43482   43483   43483   43486   43486   43487
                         43487   43489   43489   43490   43490   43492   43492   43493   43493   43502   43502   43505
                         43505   43508   43508   43511   43511   43514   43514   43517   43517   44027   44027
CM.OS.WRT-ADEC.PC        43080   43145 #
CM.OS.WRT-AUTO.DEC       43063   43155 #
CM.OS.WRT-IMED           43098   43150 #
CM.OS.WRT-ODD.CHECK1     43034   43038   43059   43106 # 43148   43153
CM.OS.WRT-ODD.CHECK1A    43084   43088   43101 #
CM.OS.WRT-ODD.CHECK2     43043   43114 #
CM.OS.WRT-ODD.CHECK2A    43118   43123 #
CM.OS.WRT-ODD.CHECK3     43048   43128 #
CM.OS.WRT-ODD.CHECK4     43069   43137 # 43159
CM.OSR-GET.IND.1         42764   42913   43054   43163 #
CM.OSR-GET.IND.2         42785   42935   43076   43175 #
CM.OSR-GET.IND.3         42804   42955   43094   43195 #
CM.OSR-GET.IND.TST       43178   43186   43199   43204   43209   43211 #
CM.OSR-IND.DEC.PC        43182   43188 #
CM.OSR-MOD.END           43232   43249 #
CM.OSR-RED.END           43227   43240 #
CM.OSR-WRT.END           43237   43259 #
CM.OSR.IND-TEST.FLAG     43169   43192   43216   43223 #
CM.ROL-MEM               41617 # 43975   43975   43978   43978
CM.ROL-PC                41604   41610 #
CM.ROL-REG               41585 # 43974   43974   43977   43977
CM.ROL-TEST              41597 #
CM.ROR-MEM               41639 # 43982   43982
CM.ROR-PC.RESULT         41698   41716 #
CM.ROR-PC.WRITE          41719   41724 #
CM.ROR-REG               41676 # 43981   43981
CM.ROR-REG.RESULT        41693   41706 #
CM.ROR-REG.TESTC         41679   41686 #
CM.ROR-WRITE             41668   41653 # 41669   41673   41709   41713
CM.RORB-MEM              41660 # 43985   43985
CM.RORB-REG              41681 # 43984   43984
CM.RTI                   42640 # 43987   43987   43988   43988
CM.RTI-TEST.PC           42664 # 42674
CM.RTI.CONTINUE          42649 # 42720
CM.RTS                   42424 # 43990   43990   43991   43991
CM.RTS-REST.REG          42435   42443   42445 #
CM.RTS.CONTINUE          42432 # 42716
CM.RTT                   42641 # 43993   43993   43994   43994
CM.SBC-MEM               41536 # 43997   43997   44000   44000
CM.SBC-REG               41517 # 43996   43996   43999   43999
CM.SOB                   42475 # 44002   44002   44003   44003   44005   44005   44006   44006
CM.SUB-MEM               41882 # 44009   44009   44012   44012
CM.SUB-REG               41888 # 44008   44008   44011   44011
CM.SWAB-MEM              41753 # 44015   44015   44018   44018   44021   44021   44024   44024
CM.SWAB-PC               41767   41779 #
CM.SWAB-REG              41740 # 44014   44014   44017   44017   44020   44020   44023   44023
CM.SWAB-RUN              41750   41759 #
CM.SWAB-SET.CC           41763   41772   41773 #
CM.SXT                   41558 # 44026   44026   44027   44027
```

Cross Reference Listing - Field Names and Defined Values

```
CM.TEST.PC              41490   41529   41614   41728   41787   41807   41826   41869   41901   42217   42259   42324
                        42398   42439   42465   42626   42667   42676 #
CM.TEST.PC.ODD          41032   41070   41181   42451   42680 #
CM.TST-MEM              41152 # 44043   44043   44046   44046
CM.TST-REG              41133 # 44042   44042   44045   44045
CM.XOR                  42186 # 44048   44048   44049   44049   44051   44051   44052   44052
CM.XOR-WRITE            42202 # 44048   44048   44049   44049   44051   44051   44052   44052
CM.XOR-WRITE1           42207 # 42226
CN.ADJUST.CHAR          8041    8048 #
CN.API.INT              8060 #
CN.B.SWITCH             7349    7478 #
CN.BOOT.CHK.DEV         7029 #  7042
CN.BOOT.CHK.DEV.10      7053    7039 #
CN.BOOT.DEV             6474    6496 #
CN.BOOT.DEV.B           7059    7068 #
CN.BOOT.DEV.C           7071    7080 #
CN.BOOT.DE..D           7083    7092 #
CN.BOOT.DEV.TYPE.OK     7066    7078    7090    7102    7104 #
CN.BOOT.WITH.DEV        7014    7019 #
CN.CHECK.DONE           8020    8028 #
CN.CHKSUM.CHK           8063    8080    8085 #
CN.CLEAR.HALT           6962 #
CN.CLEAR.TEMPS          6171    6270    8161 #  8178
CN.CLEAR.TEMPS.10       8164    8171 #
CN.CNTRL.P.HALT         6125 #
CN.CONSOLE              5366    5385    5403    5472    5504    5535    20969 #
CN.DEPOSIT.GPR          6880    6920 #
CN.DEPOSIT.PHY          6872    6876    6893 #
CN.DEPOSIT.PSL          6682    6929 #
CN.DEPOSIT.SIZE         6896    6898 #
CN.DO.BOOT              6675    7011 #
CN.DO.COMMAND           6483    6645 #
CN.DO.CONTINUE          6662    6667    6949 #
CN.DO.DEPOSIT           6653    6855 #
CN.DO.EXAMINE           6648    6741 #
CN.DO.START             6671    6954 #  7138
CN.DO.X                 6685    7158 #
CN.E.GPR                6761    6798 #
CN.E.GPR.IPR            6809 #  36174
CN.E.IPR                6765    6820 #
CN.E.OR.D.STAR          7726    7731 #
CN.E.PSL                6679    6834 #
CN.EMPTY.BUF            7695    7708 #
CN.ERROR                7037    7095    7170    7235    8184 #  35670   36184   37672
CN.EXAMINE              6753    6767 #
CN.EXAMINE.10           6773    6776 #
CN.G.SWITCH             7334    7435 #
CN.GET.BOOT.DEV         6503    6578 #  6608    6621
CN.GET.BOOT.DEV.05      6583 #
CN.GET.BOOT.DEV.10      6592    6605 #
CN.GET.BOOT.DEV.20      6587    6610 #
CN.GET.BOOT.DEV.25      6604    6615 #
CN.GET.CHAR             7162    7846    8006 #
CN.GET.CHAR.NO.ECHO     7220    8011 #
CN.GET.NEXT             6261    6456    6581    7306    7607    7844 #
```

```
CN.GET.NEXT.NOT.RO      7854    7866 #
CN.GET.NIBBLE           7636 #
CN.GET.NUM.10           7676    7681 #
CN.GET.NUM.9            7615    7624 #
CN.GET.NUM.A            7627    7632 #
CN.GET.NUM.BUFO         7644    7661 #
CN.GET.NUM.F            7635    7673 #
CN.GET.NUM.OVFLW        7641 #  7690
CN.GET.NUM.PLUS         7746    7759 #
CN.GET.NUM.PSL          7718    7743 #
CN.GET.NUM.RO.5         7782    7807 #
CN.GET.NUM.RO.BCD.CHK   7801    7820    7826 #
CN.GET.NUM.RUBOUT       6523    6552    7611    7779 #
CN.GET.NUM.RUBOUT.1     7565    7784 #
CN.GET.NUMBER           6513    6542    7553    7605 #  7621    7652    7659    7664    7671    7706    7741    7757
                        7762    7769
CN.GET.NUMBER.END       7630    7679    7683    7692 #
CN.GET.SWITCH           6491    7304 #  7383    7405    7410    7433    7438    7454    7459    7476    7499    7504
                        7521    7526    7543    7548    7578    7595
CN.GNC.CRAR.GETS.0      8022 #
CN.H.SWITCH             7378    7482    7545 #
CN.H.SWITCH.10          7550 #  7573
CN.I.SWITCH             7342    7456 #
CN.IRD1                 6981 #  6991
CN.IS.CHAR.READY        8014    8016 #  8032
CN.IS.IT.LOWER.CASE     8058    8065 #
CN.L.SWITCH             7365    7523 #
CN.NEW.LINE             6258 #  6265    6293    6307    6321    6335    6349    6363    6377    6391    6405    6419
                        6433
CN.P.SWITCH             7318    7380 #
CN.PARSE.ADDRESS        6499    6505 #
CN.PARSE.COMMAND        6293    6303    6317    6331    6345    6359    6373    6387    6401    6415    6454 #  6487
                        6499    6528    6536    6557    6596    6613    6858    6932    7393    7420    7445    7466
                        7484    7511    7533    7585    7734    7750    7765
CN.PARSE.DATA           6508    6533 #
CN.PARSE.GET.DATA       6538 #  6560
CN.PARSE.NUMBER         6510 #  6531
CN.PARSE.RECURSE        6461 #  6520    6548    7561
CN.PARSE.RUBOUT.DATA    6545    6554 #
CN.PARSE.RUBOUT.SPACE   6517    6525 #
CN.PARSE.SWITCHES       6467    6489 #
CN.PROMPT               6219 #  6428    6657    6904    6911    6918    6926    6936    7211    7239    7295    8104
                        8202    35651   35660
CN.PROMPT.10            6236    6253 #
CN.RECEIVE.CHAR         8026    8038 #
CN.RESET.STEPC          6695    6705 #
CN.RESTORE.MME          6978    6985 #
CN.RESTORE.PC           6952    6967 #
CN.RESTORE.TEMP7        8128    8132    8136    8141 #
CN.RUB.STAR             7737    7738 #
CN.SAVE.PSL             6178    6692 #
CN.STRIP.PAR            8052    8055 #
CN.TD.CHK.READY         7944    7952 #  7963
CN.TD.DEC.STEPC         7975    7980 #
CN.TD.IS.IT.READY       7955    7960 #
```

```
CN.TD.XMIT                7953    7965 #
CN.TD.XMIT.1              7971 #
CN.TRY.B                  6369    6380 #
CN.TRY.C                  6341    6352 #
CN.TRY.CNTRL.P            6425    6430 #  8046
CN.TRY.CR                 6411    6422 #
CN.TRY.D                  6285    6296 #
CN.TRY.H                  6299    6310 #
CN.TRY.I                  6397    6408 #
CN.TRY.N                  6313    6324 #
CN.TRY.S                  6327    6338 #
CN.TRY.T                  6355    6366 #
CN.TRY.X                  6383    6394 #
CN.TYPE.CHAR              6149    6160    6225    6256    6840    6974    7177    7199    7208    7264    7268    7272
                          7276    7284    7861    7873    7920    7941 #  7982    7986    8009    8094    8187    8191
                          46152   46156   47398   47412   47423
CN.TYPE.CHAR.RET          7947 #
CN.TYPE.CTRLU             8083    8091 #
CN.TYPE.DATA              6843    7281 #
CN.TYPE.NUMBER            6156    6167    7280    7291    7893 #  8195
CN.TYPE.NUMBER.10         7912    7917 #
CN.TYPE.NUMBER.CONVERT    7900    7905 #
CN.TYPE.NUMBER.LOOP       7897 #  7925
CN.TYPE.PC                6130 #  6437    20976
CN.TYPE.RESULT            6783    6791    6796    6818    7261 #  8206
CN.V.SWITCH               7326    7407 #
CN.VIRT.TO.PHY            6757    8118 #
CN.W.SWITCH               7357    7501 #
CN.X.DUMP                 7196 #  7204
CN.X.DUMP.10              7193    7201 #
CN.X.LOAD                 7184    7213 #
CN.X.LOAD.LOOP            7217 #  7227
CN.X.SWITCH               7373    7575 #
CO.ACB-REG                19327 # 19748   19748   19748   19748   19755   19755   19755   19755   19762   19762   19762
                          19762
CO.ACB-SETCC              19284   19288 #
CO.ACB-TEST.ADD.REG       19337   19343   19346 #
CO.ACB-TEST.ADDEND        19293   19299   19300 #
CO.ACB-TEST1              19304   19307 #
CO.ACB-TEST2              19311   19316   19319 # 19354   19359
CO.ACBB-MEM               19274 # 19748   19748
CO.ACBB-REG               19338 #
CO.ACBL-MEM               19276 # 19755   19755
CO.ACBL-REG               19333 #
CO.ACBW-MEM               19275 # 19762   19762
CO.ACBW-REG               19339 #
CO.AOB-BRA                19400   19407 #
CO.AOB-MEM                19391   19402 #
CO.AOB-REG.TEST           19382   19386 #
CO.AOBLEQ                 19379 # 19768   19768   19768   19768   19769   19769   19769   19769   19769   19769
CO.AOBLSS                 19378 # 19775   19775   19775   19775   19776   19776   19776   19776   19776   19776
CO.BB-FAULT               19154   19158 #
CO.BB-GET-BASE            19048   19061   19071   19140 #
CO.BB-MEM-FAULT           19093   19098   19100 #
CO.BB-MEM.FIXPC           19083   19087   19090 #
```

Cross Reference Listing - Field Names and Defined Values

```
CO.BB-MEM.IRQ       19042   19053   19077   19080 #
CO.BB-REG           19151 # 19782   19782   19782   19782   19789   19789   19789   19789   19795   19795   19795
                    19795   19802   19802   19802   19802   19809   19809   19809   19809   19816   19816   19816
                    19816   19823   19823   19823   19823   19830   19830   19830   19830
CO.BB-REG.EXIT      19184   19189   19191 #
CO.BB-REG.FIXPC     19174   19178   19181 #
CO.BB-REG.IRQ       19167   19171 #
CO.BB-RET           19146 # 19783   19783   19783   19783   19783   19783   19790   19790   19790   19790   19790
                    19790   19796   19796   19796   19796   19796   19796   19803   19803   19803   19803   19803
                    19803   19810   19810   19810   19810   19810   19810   19817   19817   19817   19817   19817
                    19817   19824   19824   19824   19824   19824   19824   19831   19831   19831   19831   19831
                    19831
CO.BBC              19045 # 19782   19782
CO.BBCC             19058 # 19789   19789
CO.BBCCI            19068 # 19795   19795
CO.BBCS             19056 # 19802   19802
CO.BBI-READ         19065   19073 #
CO.BBS              19044 # 19809   19809
CO.BBSC             19057 # 19816   19816
CO.BBSS             19055 # 19823   19823
CO.BBSSI            19067 # 19830   19830
CO.BLB-TEST         19237   19242   19246 #
CO.BLBC             19239 # 19882   19882   19882   19882   19882   19882
CO.BLBS             19234 # 19889   19889   19889   19889   19889   19889
CO.BRB              18924 # 19925   19925   19925   19925
CO.BRCND            18864 # 19835   19835   19835   19835   19842   19842   19842   19842   19848   19848   19848
                    19848   19857   19857   19857   19857   19866   19866   19866   19866   19873   19873   19873
                    19873   19894   19894   19894   19894   19900   19900   19900   19900   19907   19907   19907
                    19907   19916   19916   19916   19916   19952   19952   19952   19952   19959   19959   19959
                    19959
CO.BRCND-BRANCH     18912 # 18927   18931   18935   18939   18943   18947   18951   18955   19010   19645
CO.BRCND-DECIDE     18868   18873   18878   18883   18888   18893   18898   18903   18908 #
CO.BRW              18967 # 19932   19932   19932   19932
CO.BRW-ALLIGN       18971   18976   18981   18986   18991   18996   19001   19005   19007 #
CO.BSB              19607   19611   19615   19619   19623   19627   19631   19635   19637 # 19702
CO.BSBB             19604 # 19939   19939   19939   19939
CO.BSBW             19655 # 19946   19946   19946   19946
CO.BSBW-ALLIGN      19659   19664   19669   19674   19679   19684   19689   19693   19696 #
CO.CASE-BAD         19534   19569 #
CO.CASE-BAD.FIXPC   19572   19580 #
CO.CASE-BRANCH      19562 #
CO.CASE-DECIDE      19529   19531 #
CO.CASE-DISP.ADD    19522 # 19559
CO.CASE-GOOD        19540   19546   19561 #
CO.CASE-SETCC       19566   19583   19586 #
CO.CASEB-MEM        19509 # 19969   19969
CO.CASEB-REG        19549 # 19969   19969   19969   19969
CO.CASEL-MEM        19511 # 19976   19976
CO.CASEL-REG        19551 # 19976   19976   19976   19976
CO.CASEW-MEM        19510 # 19983   19983
CO.CASEW-REG        19550 # 19983   19983   19983   19983
CO.JMP              19019 # 19739   19989   19989   19989   19989   19989   19989
CO.JSB              19713 # 19996   19996   19996   19996   19996   19996
CO.NOP              5480    18907 # 18915   19022   19250   19254   19322   19410   19414   19418   19461   19465
                    19474   19493   41007   41045   41083   41121   41156   41193   41215   41282   41568   41573
```

Cross Reference Listing - Field Names and Defined Values

```
                          41628   41657   41837   41854   41886   41917   42211   42222   42243   42309   42538   42546
                          42551   42566   42571   42621   42631
CO.RSB                    19731 # 20001   20001   20001   20001
CO.SUB-BRA                19451   19458 #
CO.SOB-REG                19453 # 45818
CO.SOB-TEST.MOST.POS      19470   19479   19484   19487 #
CO.SOBGEQ                 19441 # 20009   20009   20009   20009   20009   20009
CO.SOBGTR                 19442 # 20016   20016   20016   20016   20016   20016
CR.CONT                   17489 # 17536
CR.CONT1                  17497 # 17626
CR.CONT2                  17500   17519 #
CR.CRC                    17439 # 17707   17707   17707   17707   17707   17707
CR.CRC.10                 17446 # 17708   17708   17708   17708   17708   17708
CR.CRC.RES                17599 # 17704   17704   17704   17704
CR.CRC.RES.10             17602   17606 #
CR.CRC.RES.20             17630   17636 #
CR.CRC.SAVE               17576 # 39011
CR.CRC.SAVE.10            17585 # 17596
CR.DONE2                  17484 # 17512
CR.DONE3                  17488   17514 #
CR.GETTBL                 17531   17544 # 17552
CR.INNER                  17525 # 17541   17640
CR.SETPLOFF               17463   17560 # 17610
CS.C3                     22465 # 24541
CS.C3.1                   22468   22478   23667 #
CS.C3.CLRFPD.1            23877   23892 #
CS.C3.LOP                 23670   23723 # 23885   25114
CS.C3.LOP.B               23731   23735   23754 # 23787
CS.C3.LOP.B.A             23757   23764 #
CS.C3.LOP.B.INT           23728 # 23818   23839
CS.C3.LOP.B1              23768   23873 #
CS.C3.LOP.L               23726   23780 #
CS.C3.LOP.L.INT           23835   23842 #
CS.C3.LOP.L1              23783   23801 # 23845   23849
CS.C3.LOP.L1.A            23804   23809 #
CS.C5                     23694   23919 # 25126
CS.C5.S1F                 23921   23955 #
CS.C5.S1F.B               23976 # 24015
CS.C5.S1F.B1              23972 # 24005
CS.C5.S1F.BC              23980   23999 # 24041
CS.C5.S1F.L               23985   24012 # 24048
CS.C5.S1F.L.INT           24036   24045 #
CS.C5.S1F.LOP             23967 # 24367
CS.C5.S1F.LOP.-1          23970   23982 #
CS.C5.S1F.R1_PC-1         23993   24002   24008 #
CS.C5.S2F                 23928   23941 #
CS.C5.S2F.1               23947   24059 # 25123
CS.C5.S2F.B               24080 # 24103
CS.C5.S2F.B1              24076 # 24093
CS.C5.S2F.BC              24086 # 24130
CS.C5.S2F.L               24074   24100 # 24136
CS.C5.S2F.L.INT           24125   24133 #
CS.C5.S2F.RO_C            24078   24089   24096 # 24112
CS.CMP.CLRFPD             23677 # 23894   23909   23925   24098
CS.CMP.PACK               23761   23808   39031 #
```

Cross Reference Listing - Field Names and Defined Values

```
CS.CMP.PACK.1              24930   24938   24941 #
CS.CMP.PACK.2              24977 # 25002
CS.CMP.PACK.3              24989   24993   24995 #
CS.CMP.PACK.BEGIN          23744   23748   23752   23792   23797   23858   23862   23866   24923 # 39034
CS.CMP.R1.GETS.PC          23672 # 23826   23889   23904
CS.CMP.R2.GETS.D           23685   23697 # 23937
CS.CMP.UNP.1A              25047   25054 #
CS.CMP.UNP.1B              25051   25059 #
CS.CMP.UNPACK              25013 # 25412   25412   25412   25412   25419   25419   25419   25419
CS.CMP.UNPACK.3            25057   25117 #
CS.CMP.UNPACK.4            25071   25078 #
CS.CMP.UNPACK.5            25092   25107 #
CS.CMP.UNPACK.6            25088 # 25104
CS.CMP.UNPACK.BEGIN        25020 # 25372
CS.EXP.FILLER              22533   22674   22759 # 24754   24945   25037   25162   25284
CS.FIX.MATCHC.IND          22799 # 24463   24526
CS.GEN.RESPC               22779 # 22930   23494   24443   24501   24533
CS.GEN.RESPC.1             22783 # 23681   23933
CS.LOC                     22684   24311 # 24472
CS.LOC.MAT                 24305 # 24328   24334
CS.LOC.SPL                 24314   24321 # 24343
CS.LOC.SPL.1               24324   24331 #
CS.LOCC.SKPC.PACK          39022 #
CS.LOCC.SKPC.PACK.BE       25241 # 39025
CS.LOCC.SKPC.UNPACK        25274 # 25368   25426   25426   25426   25426   25474   25474   25474   25474
CS.M3                      22463   22471 # 22482   22486
CS.M3.1                    22474   22910 #
CS.M3.MTUC                 22913   23351 #
CS.M3.R1_PC                23120   23208 #
CS.M3.WR                   23075   23083 # 23116   24821
CS.M3.WR.B                 23088 # 23133
CS.M3.WR.B.A               23091   23100 #
CS.M3.WR.B.B               23103   23108 #
CS.M3.WR.BACK              23001   23015 # 23061   24847
CS.M3.WR.BACK.1            23018   23022   23043 #
CS.M3.WR.BACK.1A           23046   23051 #
CS.M3.WR.L                 23086   23126 #
CS.M3.WR.L.LOP             23152 # 23197
CS.M3.WR.L.LOP1            23155   23159   23180 #
CS.M3.WR.L.LOP1.A          23183   23188 #
CS.M3.WR.L1                23129   23135 #
CS.M3.WR.LB                23131 # 23215
CS.M5                      22940   23228 # 23570
CS.M5.WR                   23251 # 23266   24810
CS.M5.WR.B                 23256 # 23283
CS.M5.WR.L                 23254   23276 #
CS.M5.WR.L.LOP             23304 # 23325
CS.M5.WR.L.LOP1            23307   23311   23317 #
CS.M5.WR.L1                23279   23286 #
CS.M5.WR.LB                23281 # 23343
CS.MAT.EXIT                24440 # 24489
CS.MAT.Q.GETS.Q-R0         23704   24544 #
CS.MAT.R0.Q.D              24309   24492 #
CS.MAT.R0.Q.D.1            24495   24513 #
CS.MAT.R1.GETS.R1-Q        24547   24563 #
```

Cross Reference Listing - Field Names and Defined Values

```
CS.MAT.R1.GETS.VA.CL      24504 # 24552
CS.MAT.R2.GETS.10-Q       24509   24555 #
CS.MAT.R3.GETS.TMP2       24537   24549 #
CS.MAT.SET.FPDOFF         22731   24483 #
CS.MAT.SET.FPDOFF.-1      24420 # 24486   24576   25383
CS.MAT.TMP5               24454 # 24481
CS.MAT.TMP5.-1            24423   24475 #
CS.MAT.UNP.1              25364   25374 #
CS.MAT.UNP.L.S            25308   25316 #
CS.MATCHC.PACK            39026 #
CS.MATCHC.UNPACK          25355 # 25433   25433   25433   25433
CS.MDR.MASK.SCAN          24216   24223 #
CS.MDR.MASK.SPAN          24221   24254 #
CS.MOV.BUT.FLAG1          23211   23336 #
CS.MOV.PACK               23050   23107   23187   39036 #
CS.MOV.PACK.1             24658 # 24665
CS.MOV.PACK.2             24661   24667 #
CS.MOV.PACK.BEGIN         23031   23035   23039   23096   23168   23172   23176   24627 # 39039
CS.MOV.R4.GETS.0          22926 # 23065   23231   23270   23339   23354
CS.MOV.R5.GETS.0          22936   22944 #
CS.MOV.UNPACK             24733 # 25440   25440   25440   25440   25447   25447   25447   25447   25454   25454   25454
                          25454   25460   25460   25460   25460
CS.MOV.UNPACK.1           24770   24776 #
CS.MOV.UNPACK.3           24802   24813 #
CS.MOV.UNPACK.4           24816   24822 #
CS.MOV.VA.END.IN.00       22965   22981   23071 #
CS.MOVT.DEC.PC            23532   23548 #
CS.MOVT.DEC.VA            23551   23560 #
CS.MOVT.R5.GETS.VA        23490 # 23515
CS.MOVTC.TUC.BACK         23008   23434 #
CS.MOVTC.TUC.FIN          23219 # 23558
CS.MOVTC.TUC.TMP10-0      23222   23567 #
CS.MTC.FOB                23475   23524 #
CS.MTC.FOB.PRE            23465   23518 #
CS.MTC.R2_0.IRD1          22643   23507 # 24561
CS.MTC.VA                 23450   23454   23462 #
CS.MTC.WR                 23079   23447 # 23541   23563   24869
CS.MTUC.END               23358   23481 # 23574
CS.MVT.FINISH             23545   23553 #
CS.OS.LOC                 22681 # 25314   25326
CS.OS.LOCC.SKPC           22650 # 25429   25429   25429   25429   25429   25429   25477   25477   25477   25477   25477
                          25477
CS.OS.LOCC.SKPC.1         22657 # 25430   25430   25430   25430   25430   25430   25478   25478   25478   25478   25478
                          25478
CS.OS.MATCHC              22695 # 25436   25436   25436   25436   25436   25436
CS.OS.MATCHC.1            22702 # 25437   25437   25437   25437   25437   25437
CS.OS.MOV.CMP.IN          22436   22443 #
CS.OS.MOV.CMP.INT         22433 # 22547
CS.OS.MOVC3.CMPC3         22415 # 25415   25415   25415   25415   25415   25415   25443   25443   25443   25443   25443
                          25443
CS.OS.MOVC3.CMPC3.1       22422 # 25416   25416   25416   25416   25416   25416   25444   25444   25444   25444   25444
                          25444
CS.OS.MOVC5.CMPC5         22494 # 25422   25422   25422   25422   25422   25422   25450   25450   25450   25450   25450
                          25450
CS.OS.MOVC5.CMPC5.1       22500 # 25423   25423   25423   25423   25423   25423   25451   25451   25451   25451   25451
```

Cross Reference Listing - Field Names and Defined Values

```
                                25451
CS.OS.MOVTC.MOVTUC              22554 # 25457   25457   25457   25457   25457   25457   25463   25463   25463   25463   25463
                                25463
CS.OS.MOVTC.MOVTUC.1            22559 # 25458   25458   25458   25458   25458   25458   25464   25464   25464   25464   25464
                                25464
CS.OS.MOVTC.TUC.CONT            22521 # 22586
CS.OS.R3.VA.GET.TMP2            22624   22722   22770 #
CS.OS.SCANC.SPANC               22594 # 25470   25470   25470   25470   25470   25470   25484   25484   25484   25484   25484
                                25484
CS.OS.SCANC.SPANC.1             22600 # 25471   25471   25471   25471   25471   25471   25485   25485   25485   25485   25485
                                25485
CS.OS.SCSP.BEG                  22635 # 25215
CS.PC.GETS.PC-1.SCAN            24191   24241   24248 #
CS.PC.GETS.R1                   25031   25204   25289   25386 #
CS.PCK.MOV                      24638 # 24651
CS.PCK.MOV.1                    24641   24653 #
CS.R0-1.FOR                     23527   23533 #
CS.R0.D.GETS.RB+1.SC.SP         24236 # 24265
CS.R0.GETS.R0+10                23484   23512 #
CS.R1.GETS.PC                   24970   25250   25392 #
CS.R1.GETS.PC.SCAN              24243 # 24251
CS.R2.GETS.DELPC.R2             22447   22717   22743 #
CS.SC.SPAN.PACK                 39018 #
CS.SC.SPAN.PACK.BEGI            25158 # 39020
CS.SC.SPAN.UNPACK               25194 # 25467   25467   25467   25467   25481   25481   25481   25481
CS.SCSP                         22638   24193 #
CS.SCSP.EXIT                    22640 # 24246
CS.SCSP.R0.INC                  24187 # 24228   24259
CS.SCSP.SPL                     24202   24206   24213 #
CS.SK                           22688   24363 #
CS.SK.EXIT                      24369 # 24377
CS.SRCADD.VS.DESTADD            22921   22962 #
CS.UNP.CMP.CSS1                 25110   25126 #
CS.UNP.GET.R3                   24844   24860   24874 #
CS.UNP.MOV                      24839   24852 #
CS.UNP.MOV.1                    24855   24871 #
CS.UNP.MOV.MTC                  24866 # 24873
CS.UNP.TMP10.IS.NEG             24775   25076   25399 #
DS.ADDP4                        26154   26516   26575 #
DS.ADDP4.ADD                    26700   26715   26736   26741   26755 #
DS.ADDP4.ADD.SPL                26671   26675   26696 #
DS.ADDP4.FLAGS                  26637   26869 # 26884   26888   26896   27337
DS.ADDP4.FLAGS.SIGN             26875   26893 # 26900
DS.ADDP4.OPC                    26589   26601 #
DS.ADDP4.OPCODE                 26653   26668 #
DS.ADDP4.REE                    26576 # 32177
DS.ADDP4.REWR                   26627   26643 #
DS.ADDP4.SRC1                   26583   26591 #
DS.ADDP4.SRC1SGN                26604   26612 #
DS.ADDP4.SRC2                   26616   26621 #
DS.ADDP4.SRC2SIGN               26640   26650 #
DS.ADDP4.SUB                    26705   26710   26731   26746   26806 #
DS.ADDP4.SUB.SPL                26679   26683   26727 #
DS.ADDP4.WRITE                  26772   26787 # 26824
DS.ADDP6                        26415 # 33261   33261   33261   33261   33261   33261   33330   33330   33330   33330   33330
```

```
                    33330   33344   33344   33344   33344   33344   33344   33357   33357   33357   33357   33357
                    33357
DS.ADDP6.2          26433 # 33262   33262   33262   33262   33262   33262   33331   33331   33331   33331   33331
                    33331   33345   33345   33345   33345   33345   33345   33358   33358   33358   33358   33358
                    33358
DS.ADDP6.EXIT       26480   26483   26486   26494 #
DS.ADDP6.EXIT.2     26501   26507 #
DS.ADDP6.LEN        26429   26462 #
DS.ASHP             31408 # 33268   33268   33268   33268   33268   33268
DS.ASHP.2           31419 # 33269   33269   33269   33269   33269   33269
DS.ASHP.BYT.BIG     31588   31615 #
DS.ASHP.BYT.BIG.2   31618   31624 #
DS.ASHP.BYT.SHF     31575   31585 #
DS.ASHP.BYT.SPL     31554   31572 # 31604   31635
DS.ASHP.CNT         31491 # 31498
DS.ASHP.CSIGN       31494   31502 # 31509
DS.ASHP.DONE        31579   31671 #
DS.ASHP.FLAGS       28859   31715   31738 # 31745   31753   31757
DS.ASHP.INIT        31470   31477 #
DS.ASHP.INTC        31461   31467 #
DS.ASHP.LENGTH      31416   31453 #
DS.ASHP.LOOP.SGN    31523   31531 #
DS.ASHP.LOOP.SPL    31535   31546 #
DS.ASHP.NIB         31549   31647 #
DS.ASHP.NIB.BIG     31650   31657 #
DS.ASHP.PRE.READ    31505   31512 #
DS.ASHP.REE         31490 # 32163
DS.ASHP.SIGN        31675   31701 #
DS.ASHP.SIGN.2      31704   31708   31717 #
DS.ASHP.SOURCE.IN   31518   31525 #
DS.CCPCIRD1         25725   26871   26880   31740   31748   31773 #
DS.CMPP.-0+0        25997   26003   26027 #
DS.CMPP.EXIT        26019 # 28866
DS.CMPP.EXT1        26011 #
DS.CMPP.FLAGS       25932   25951 #
DS.CMPP.FLAGS.OK    25954   25992 #
DS.CMPP.REE         25792   25804 # 32082   32124
DS.CMPP.SRC1.1      25801   25815 #
DS.CMPP.SRC1.2      25818   25825 #
DS.CMPP.SRC1.2A     25822   25830 #
DS.CMPP.SRC1.3      25828   25835 #
DS.CMPP.SRC1.3A     25833   25840 #
DS.CMPP.SRC1.4      25838   25846 #
DS.CMPP.SRC1.SGN    25843   25848   25855 #
DS.CMPP.SRC2        25859   25866 #
DS.CMPP.SRC2.SIGN   25797   25882 #
DS.CMPP.SUB.1       25886   25893 #
DS.CMPP.SUB.2       25896   25904 #
DS.CMPP.SUB.2A      25901   25909 #
DS.CMPP.SUB.3       25907   25917 #
DS.CMPP.SUB.3A      25913   25923 #
DS.CMPP.SUB.LAS     25921   25929 #
DS.CMPP3            25644   25788 # 26241   26284
DS.CMPP4            26086 # 33254   33254   33254   33254   33254   33254   33282   33282   33282   33282   33282
                    33282   33350   33350   33350   33350   33350   33350
```

```
DS.CMPP4.34              26181   26221 #
DS.CMPP4.ADD4            26131   26140 #
DS.CMPP4.ADD4.1          26143   26148 #
DS.CMPP4.C1              26136   26166 #
DS.CMPP4.ELOP.1L         26271 # 26322
DS.CMPP4.ELOP.2L         26250 # 26318
DS.CMPP4.FREE2           26224   26235 # 26256
DS.CMPP4.L1              26177   26196 #
DS.CMPP4.LOP             26300 # 26330
DS.CMPP4.LOP.SPL         26303   26328 #
DS.CMPP4.PRELOP          26193   26198   26291 #
DS.CMPP4.REE             26133 # 32100
DS.CMPP4.S2              26093 # 33255   33255   33255   33255   33255   33283   33283   33283   33283   33283
                         33283   33351   33351   33351   33351   33351   33351
DS.CMPP4.S3              26111   26118 #
DS.CMPP4.UNEQ.CC2        26354   26368 #
DS.CMPP4.UNEQ.SGN        26338   26343 #
DS.CMPP4.UNEQ.SPL        26346 #
DS.CVT.PACK              39041 # 39117   39123
DS.CVTLP                 29455 # 33290   33290   33290   33290   33290   33290
DS.CVTLP.10              29467 # 29474
DS.CVTLP.100             29698   29723   29728 #
DS.CVTLP.20              29470   29476 #
DS.CVTLP.30              29495   29500   29506 #
DS.CVTLP.40              29508   29517 #
DS.CVTLP.45              29482   29522 #
DS.CVTLP.50              29520   29527 #
DS.CVTLP.70              29556   29563 #
DS.CVTLP.80              29567   29609 #
DS.CVTLP.81              29585   29590   29595   29600   29605   29613 #
DS.CVTLP.83              29579   29617 #
DS.CVTLP.85              29660   29664   29670   29677 #
DS.CVTLP.90              29645   29651   29682 #
DS.CVTLP.95              29680   29686 #
DS.CVTLP.95              29703   29712   29719 #
DS.CVTLP.CHECK.OVFLW     29552 # 29560
DS.CVTLP.DONE            29752 # 29768
DS.CVTLP.LOOP            29497 # 29530
DS.CVTLP.NON.0.DIGIT     29546   29550   29771 #
DS.CVTLP.OVFLW           29575   29582 #
DS.CVTLP.SIGN            29525   29532 #
DS.CVTLP.SRCLEN          29535   29540 #
DS.CVTLP.WRT.BYTE        29700 # 29736
DS.CVTLP.WRT.DST         29628   29634 #
DS.CVTLP.WRT.LONG        29647 # 29694
DS.CVTNP                 30202   30481   30562 #
DS.CVTNP.50              30590   30604 #
DS.CVTNP.60              30641   30662 #
DS.CVTNP.80              30691   30696   30702 #
DS.CVTNP.90              30714   30721 #
DS.CVTNP.CHK.10          30772   30778 #
DS.CVTNP.CHK.BAD         30596   30766 #
DS.CVTNP.CHK.INPUT       30681   30693 #
DS.CVTNP.CHK.OVFLW       30581   30593 #
DS.CVTNP.CONVERT         30586   30626   30638 #
```

```
DS.CVTNP.FILL              30611   30620   30628 #
DS.CVTNP.FILL.DONE         30622 # 30636
DS.CVTNP.LOOP              30665   30674   30677 # 30725
DS.CVTNP.NON.ZERO          30687 # 30700
DS.CVTNP.RETURN            30653 # 30756   30760
DS.CVTNP.SINGLE            30645   30670   30729 #
DS.CVTNP.SINGLE.10         30747   30753 #
DS.CVTNP.STORE.ZERO        30616 # 30632
DS.CVTNP.STRIP.ZEROS       30577 # 30601   30787
DS.CVTPL                   29806 # 33296   33296   33296   33296   33296   33296
DS.CVTPL.02                29812 # 33297   33297   33297   33297   33297   33297
DS.CVTPL.20                29843   29852 #
DS.CVTPL.30                29877   29889 #
DS.CVTPL.50                29931   29966 #
DS.CVTPL.CHECK.OVFLW       29850   29866 #
DS.CVTPL.CLEAR.R0          29982   29989 #
DS.CVTPL.CLR.R3            30046   30052 #
DS.CVTPL.CONVERT           29873   29882   29892   29897 # 29907
DS.CVTPL.END.CONVERT       29860   29886   29917 #
DS.CVTPL.IRD1              30017 # 30054   31313   31780
DS.CVTPL.LOOP              29879 # 29912
DS.CVTPL.NEG.OVR           29950   29971 #
DS.CVTPL.NO.OVFLW          29954   29969   29974   29979 #
DS.CVTPL.OVERFLOW          29977   29984 #
DS.CVTPL.SKIP.ZEROS        29839 # 29856
DS.CVTPL.UNPACK            30023 # 33293   33293   33293   33293
DS.CVTPL.WRT.DST           29992   29998 #
DS.CVTPN                   30105   30352   30820 #
DS.CVTPN.20                30853   30876 #
DS.CVTPN.40                30849   30881 #
DS.CVTPN.CONVERT           30837   30890 #
DS.CVTPN.CONVERT.10        31040   31044   31046 #
DS.CVTPN.CONVERT.20        31049   31055 #
DS.CVTPN.EVEN.LEN          30884   30893   30900 #
DS.CVTPN.INC.VA            30959   30965 #
DS.CVTPN.LAST3             30942   30955 #
DS.CVTPN.LOOP              30909   30928 # 30974   30978
DS.CVTPN.ODD.LEN           30906 #
DS.CVTPN.RETURN            30888   30934 # 30968   31058
DS.CVTPN.SINGLE            30904   30911 # 31062
DS.CVTPN.SINGLE.WRT        30915   30919   30921 #
DS.CVTPN.STRIP.LAST        31014   31036 #
DS.CVTPN.STRIP.LOOP        30992   30995 # 31034
DS.CVTPN.STRIP.MORE        31002   31009   31021   31026   31030 #
DS.CVTPN.STRIP.OVFLW       30998   31005 #
DS.CVTPN.STRIP.SNGLE       30984   31017 #
DS.CVTPN.STRIP.ZEROS       30833   30980 #
DS.CVTPN.WRT.LONG          30948   30970 #
DS.CVTPN.WRT.ZERO.LP       30865   30871 #
DS.CVTPN.WRT.ZEROS         30846 # 30875   30879
DS.CVTPS                   30322 # 33316   33316   33316   33316   33316   33316
DS.CVTPS.02                30329 # 33317   33317   33317   33317   33317   33317
DS.CVTPS.CONVERT           30349 #
DS.CVTPS.FIX.R3            30384   30392   30396   30398 #
DS.CVTPS.RESTART           30340 # 31249
```

Cross Reference Listing - Field Names and Defined Values

```
                                                                I  2
DS.CVTPS.SETCC          30374   30380   30382 # 30388
DS.CVTPS.WRT.SIGN       30360   30365   30367 #
DS.CVTPT                30081 # 33309   33309   33309   33309   33309   33309
DS.CVTPT.02             30087 # 33310   33310   33310   33310   33310   33310
DS.CVTPT.10             30123 #
DS.CVTPT.20             30114   30135 #
DS.CVTPT.RESTART        30102 # 31257
DS.CVTPT.SETCC          30111 #
DS.CVTSP                30425 # 33323   33323   33323   33323   33323   33323
DS.CVTSP.0.DEST         30492   30529 #
DS.CVTSP.02             30431 # 33324   33324   33324   33324   33324   33324
DS.CVTSP.20             30457   30461   30466 #
DS.CVTSP.40             30485   30493 #
DS.CVTSP.RESTART        30478 # 31253
DS.CVTSP.SETCC          30510   30516 #
DS.CVTTP                30172 # 33302   33302   33302   33302   33302   33302
DS.CVTTP.02             30178 # 33303   33303   33303   33303   33303   33303
DS.CVTTP.CHK.SIGN       30244   30259 #
DS.CVTTP.DLEN.0         30253   30293 #
DS.CVTTP.LEGAL.SIGN     30230   30235 #
DS.CVTTP.MINUS          30269 # 30277
DS.CVTTP.PLUS           30211   30263 #
DS.CVTTP.RESTART        30199 # 31261
DS.CVTTP.SETCC          30218 # 30296   30521
DS.CVTTP.ZERO.SRC       30208 #
DS.CVTXX.PACK           31161 # 39044
DS.CVTXX.SETCC          30127   30132   30138   30142   30401   31283 # 31295
DS.CVTXX.SETCC.06       30267   30286   30290   30532   30536   31288 #
DS.CVTXX.SETCC.07       31297 #
DS.CVTXX.SETCC.08       30273   31303 #
DS.CVTXX.SETCC.10       31286   31291   31307 #
DS.CVTXX.SETUP          30571   30823   31106 #
DS.CVTXX.UNPACK         31212 # 33299   33299   33299   33299   33306   33306   33306   33306   33313   33313   33313
                        33313   33320   33320   33320   33320
DS.DIV10                27023   27044   31653   32231 #
DS.DIV10.DIVP           28105   28116   32194 # 32625
DS.DIVP                 26489   27663 #
DS.DIVP.BAD             27770   28863 # 29121
DS.DIVP.CC              28568   28762 #
DS.DIVP.CC.DRSGN        28793   28803 #
DS.DIVP.CC.DVSGN        28770   28779   28786 #
DS.DIVP.CC.EXIT         28810   28824   28828   28832   28844 #
DS.DIVP.CC.QNP          28806   28821 #
DS.DIVP.DIGSHF          28091   28114 #
DS.DIVP.DRDV            27855   27882 #
DS.DIVP.DRSGN           27724   27734 #
DS.DIVP.DV.SHF          28070   28084 #
DS.DIVP.DVIN.1          27753   27785 #
DS.DIVP.DVIN.DIG        27797   27811 #
DS.DIVP.DVIN.INT        27762 # 27788
DS.DIVP.IN1             27683   27690 #
DS.DIVP.IN1A            27680 # 27687
DS.DIVP.IN2             27701   27707 #
DS.DIVP.ML.DIGIT        28490   28504 #
DS.DIVP.ML.OVC          28607   28637   28641   28670   28684 #
```

```
DS.DIVP.ML.Q              28538   28580 #
DS.DIVP.ML.Q1             28583   28630 #
DS.DIVP.ML.S              28428   28455 # 28511
DS.DIVP.ML.S.CON          28462   28520 #
DS.DIVP.ML.S.HIG          28534   28552 #
DS.DIVP.ML.S.L            28458   28467 #
DS.DIVP.ML.SHIFT          28559   28574   28687   28701 #
DS.DIVP.ML.SKIP           28470   28482 #
DS.DIVP.ML.TO             28451 # 28750
DS.DIVP.ML.ZER            28594   28619   28623   28633   28658 #
DS.DIVP.ML.ZER.1          28661   28667 #
DS.DIVP.ML.ZWR            28598   28615 #
DS.DIVP.MT.FILL           27890   27935 #
DS.DIVP.MT.FILL.3B        27946   27977 #
DS.DIVP.MT.FILL.3B.1      27980   27987 #
DS.DIVP.MT.FILL.L         27950   28001 #
DS.DIVP.MT.FILL.L.1       28004   28011 #
DS.DIVP.MT.FILL.L1        28018   28025 #
DS.DIVP.MT.FILL.W         27942   27957 #
DS.DIVP.MT.LOAD           27938   27965   27996   28036 # 28079
DS.DIVP.MT.LOAD.INT       28041 #
DS.DIVP.MT.LOAD.LW        28061   28073 #
DS.DIVP.MT.LOAD.SPL       28039   28058 #
DS.DIVP.MT.SHIFT          28103 #
DS.DIVP.MT1               27839   27866 #
DS.DIVP.MT2               27849   27870 # 27916
DS.DIVP.MT3               27859   27874 #
DS.DIVP.PCK.12            27720   27775   27779   27968 # 27984   28008   28049   28065   28183   28188
DS.DIVP.PCK.12.CON        27971   28022   28939 #
DS.DIVP.PCK.12.PRE        27718 # 27732
DS.DIVP.PCK.34            28218   28969 #
DS.DIVP.PCK.4             28312   28951 #
DS.DIVP.PRE.STACK         27923   28119 #
DS.DIVP.PRE.STACK.1       28110   28123 #
DS.DIVP.QFIX              28167   28205 #
DS.DIVP.QFIX.1            28208   28238 #
DS.DIVP.QFIX.FIL          28279   28294 # 28334
DS.DIVP.QFIX.FIL.CK       28285   28320 #
DS.DIVP.QFIX.FIL.CK1      28324   28331 #
DS.DIVP.QFIX.FLAGS        28253   28261 #
DS.DIVP.QFIX.PRE.SS       28300 # 28338
DS.DIVP.QFIX.SPL          28264   28275 #
DS.DIVP.REE.0             28414 # 29167
DS.DIVP.REE.1             27698 # 29185
DS.DIVP.REE.2             27735 # 29238
DS.DIVP.REE.3             28206 # 29258
DS.DIVP.REE.4             28276 # 29285
DS.DIVP.SS                28303   28307   28392 #
DS.DIVP.STACK             28144   28149 #
DS.DIVP.STACK.INT         28155   28174 #
DS.DIVP.STACK.LOP         28157 # 28172
DS.DIVP.STACK.WR          28160   28164 #
DS.DIVP.TMPSPL            27801   27814   27825 #
DS.DIVP.TMPSPL.1          27835   27842 #
DS.DIVP.TMPSPL.2          27845   27852 #
```

```
DS.DIVP.UNP                29072 # 33327   33327   33327   33327
DS.DIVP.UNP.0              29130   29152 #
DS.DIVP.UNP.1              29134   29174 #
DS.DIVP.UNP.2              29138   29344 #
DS.DIVP.UNP.2.END          29224   29232 #
DS.DIVP.UNP.3              29142   29244 #
DS.DIVP.UNP.4              29146   29266 #
DS.DIVP.UNP.CO.034         29156   29250   29269   29332 #
DS.DIVP.UNP.CO.034.C       29366   29374 #
DS.DIVP.UNP.CO.034EX       29392   29419 #
DS.DIVP.UNP.CO.034LC       29398   29407 #
DS.DIVP.UNP.CO.034LO       29389 # 29416
DS.DIVP.UNP.CO.034LP       29381   29395 #
DS.DS.PCK.CMPP34           31899 # 39091
DS.DS.PCK.CMPP4            31918 # 39083
DS.DS.PCK.GPR0             31875   39076 #
DS.DS.PCK.GPR0.CON         31851 # 31915   39079
DS.DS.PCK.GPR0.CON.A       31855 # 31902
DS.DS.PCK.GPR2.2-3         31869 # 31893   39067
DS.DS.PCK.GPR2.3CON        31879 # 39071
DS.DS.PCK.MULP             31936 # 39075
DS.DS.PCK.MULP.EX          31862 # 31948
DS.DS.UNP                  32027 # 33251   33251   33251   33251   33258   33258   33258   33258   33265   33265   33265
                           33265   33272   33272   33272   33272   33279   33279   33279   33279   33334   33334   33334
                           33334   33341   33341   33341   33341   33347   33347   33347   33347   33354   33354   33354
                           33354
DS.DS.UNP.2                32055   32059   32072   32129 #
DS.DS.UNP.CMPP34           32068   32078 #
DS.DS.UNP.CMPP4.0          32090   32109 #
DS.DS.UNP.NASH             32140   32170 #
DS.INC.VA.BY.2             30857   30861   31346 #
DS.MOVP                    25598 # 33275   33275   33275   33275   33275   33275   33337   33337   33337   33337   33337
                           33337
DS.MOVP.2                  25605 # 33276   33276   33276   33276   33276   33276   33338   33338   33338   33338   33338
                           33338
DS.MOVP.2.PSLC             25626   25632 #
DS.MOVP.EXIT               25710   25723 # 25728
DS.MOVP.FLAGS              25687   25707 # 25714   25718
DS.MOVP.INIT               25630   25637 #
DS.MOVP.INIT.2             25640   25645 #
DS.MOVP.REE                25646 # 32064
DS.MOVP.SIGN               25656   25674 #
DS.MOVP.STATE.INT          25650   25658 #
DS.MOVP.WRITE              25677   25681   25689 #
DS.MUL10                   27357   27361   31659   31662   32268 #
DS.MULP                    26608   27005 #
DS.MULP.2                  27014   27020 #
DS.MULP.END                27098   27286 #
DS.MULP.FINISH             27293   27313 #
DS.MULP.FINISH.1           27316   27323 #
DS.MULP.FINISH.2           27333   27354 #
DS.MULP.FINISH.SIGN        27344   27372 #
DS.MULP.LOP1               27110 # 27272
DS.MULP.LOP1.0             27161   27193 #
DS.MULP.LOP1.1             27165   27204 #
```

```
DS.MULP.LOP1.2          27169   27214 #
DS.MULP.LOP1.23         27217   27231 #
DS.MULP.LOP1.3          27173   27224 #
DS.MULP.LOP1.ALUS       27197   27207   27252 #
DS.MULP.LOP1.END        27137   27179   27183   27187   27259   27269 #
DS.MULP.LOP1.MUL        27118   27122   27126   27134 #
DS.MULP.LOP2            27084 # 27304
DS.MULP.LOP2.2          27087   27095 #
DS.MULP.LOP2.LOOP       27289   27296 #
DS.MULP.MUL             27141   27415 #
DS.MULP.MUL.DD          27436   27444 #
DS.MULP.MUL.EXIT        27442   27463 #
DS.MULP.MUL.EZER        27428   27439 #
DS.MULP.MUL.LOP         27425 # 27455
DS.MULP.READ            27039   27046 #
DS.MULP.SRC2BIN         27102   27492 #
DS.MULP.TOP             27063   27069 #
DS.READ                 25661   25806   25869   25873   26593   26645   27048   31527   32369 # 32375
DS.READ.2               32372   32382 #
DS.READ.3B              32404   32472 #
DS.READ.3B.1            32475   32480 #
DS.READ.3B.2            32491   32496 #
DS.READ.BYT             32396   32432 #
DS.READ.BYT.1           32435   32440 #
DS.READ.DIVP.DR         27705   29214   29556   32577 #
DS.READ.EX              32446   32467   32502   32533 #
DS.READ.EX.D            32556   32584 #
DS.READ.EX.D.0          32591   32605 #
DS.READ.EX.D.1          32595   32609 #
DS.READ.EX.D.2          32599   32613 #
DS.READ.EX.D.EX         32639   32671   32702   32717 #
DS.READ.EX.D.EX.F       32720   32725 #
DS.READ.EX.D.LOP        32632 # 32687
DS.READ.EX.D.LOP.END    32664   32680 #
DS.READ.EX.D.LOP1       32635   32642 #
DS.READ.EX.D.LOP2       32645   32652 #
DS.READ.EX.D.LOP3       32655   32661 #
DS.READ.EX.D.LT16       32659   32704 #
DS.READ.EX.D.LT8        32649   32699 #
DS.READ.EX.D.SET        32603   32622 #
DS.READ.EX.R            32551   32561 #
DS.READ.EX.SPL          32537   32548 #
DS.READ.LONG            32408   32523 #
DS.READ.LOP             32393 # 32518
DS.READ.MID             32412   32509 #
DS.READ.WRD             32400   32453 #
DS.READ.WRD.1           32456   32461 #
DS.SR.123SWP            32196   32211   33227 #
DS.SR.18                27960   28013   33144 #
DS.SR.18.CON            33147   33153 #
DS.SR.21                28406   28979   33162 #
DS.SR.22                28718   33180 #
DS.SR.23                28730   33188 #
DS.SR.24                28738   33196 #
DS.SR.25                28714   28734   33205 #
```

```
DS.SR.26                29181   29235   29254   29281   33126 #
DS.SR.28                28941   33169 #
DS.SR.456SWP            32233   32248   32271   32284   33213 #
DS.SR.BIN.MUL10.        27503   27513   27517   27526   27530   27534   33113 #
DS.SR.BIN.MUL10..2      27499   33117 #
DS.SR.GPROIN.FLAGS      28402   33242 #
DS.TABLE.ADDR           30109   30206   31331 #
DS.WRITE                25692   26790   26794   26848   26853   27375   27379   31725   31728   32788 #
DS.WRITE.3B             32837   32989 #
DS.WRITE.3B.NZ          32992   32999 #
DS.WRITE.3B.OUT         33002   33009 #
DS.WRITE.BYT            32829   32876 #
DS.WRITE.BYT.EV         32879   32900 #
DS.WRITE.BYT.EX         32889   32894   32913 #
DS.WRITE.BYT.OUT        32916   32923 #
DS.WRITE.EXIT           32926   32930   32979   33047   33058 #
DS.WRITE.EXIT.OV        33061   33071 #
DS.WRITE.EXIT.OV.6      33078   33083   33091 #
DS.WRITE.LNG            32840   33024 #
DS.WRITE.LNG.EV         33027   33034 #
DS.WRITE.LNG.EX         33031   33044 #
DS.WRITE.LOP            32820 # 32867   32871
DS.WRITE.LOP.1          32815   32826 #
DS.WRITE.MID            32844   32855 #
DS.WRITE.MID.2          32858   32865 #
DS.WRITE.WRD            32633   32935 # 33019
DS.WRITE.WRD.EV         32938   32945 #
DS.WRITE.WRD.EX         32965   32970   32976 # 32984
DS.WRITE.WRD.OUT        32942   32948   32961 #
ED.00-04                33643   33709 #
ED.40-47                33624   33774 #
ED.ADJUST.INPUT         33814   33990 #
ED.ADJUST.LOOP          33994   34001 # 34038
ED.ADJUST.OVERFLOW      33996 # 34022
ED.ADJUST.P1            34005   34015 # 34733
ED.ADJUST.SET.CC        33999   34031 #
ED.BLANK.ZERO           33806   33891 #
ED.BLANK.ZERO.LOOP      33896 # 33904   34755
ED.BLANK.ZERO.PACK      33908   33914 #
ED.DONE.OS.EVAL         33551 #
ED.EDITPC               33463 # 347.    34775   34775   34775   34775   34775
ED.EDITPC.02            33480 # 34776   34776   34776   34776   34776   34776
ED.EDITPC.10            33515   33520 #
ED.END.FLOAT            33716   33735 #
ED.EXIT                 33711   34533 #
ED.EXIT.10              34563 # 34570
ED.FILL                 33669   34065 # 34748
ED.FILL.LOOP            34072   34078 # 34089
ED.FILL.P2              34082 # 34751
ED.FLOAT                33677   34235 #
ED.FLOAT.100            34277 #
ED.FLOAT.CHK.PSLC       34275   34283 #
ED.FLOAT.LOOP           34243 # 34260
ED.FLOAT.P1             34237   34247 # 34729
ED.FLOAT.P2             34251 # 34745
```

```
ED.FLOAT.SIGNIF          34256   34271 #
ED.FLOAT.STORE           34280   34291   34312 #
ED.FLOAT.STORE.SIGN      34287   34307 #
ED.INSERT                33802   33856 #
ED.LOAD.SIGN             33788   33793   33817 #
ED.LOOP                  33559   33569 # 33722   33728   33742   33783   33823   33828   33836   33894   33912   33942
                         34012   34025   34080   34137   34245   34510   34762
ED.MINUS                 33535   33543 #
ED.MOVE                  33673   34127 #
ED.MOVE.CLEAR.Z          34178   34183 #
ED.MOVE.LOOP             34135 # 34150
ED.MOVE.P1               34129   34139 # 34725
ED.MOVE.P2               34143 # 34737
ED.MOVE.PACK             34029   34039 # 34268
ED.MOVE.SIGNIF           34147   34155 #
ED.MOVE.STORE            34181   34190 #
ED.MOVE.STORE.DIGIT      34187   34195 #
ED.MOVE.UNCONVERT        34154   34165 # 34264
ED.NOT.MINUS             33799   33826 #
ED.PACK                  33579   33633   33917   34076   34093   34162   34168   34172   34305   34505   34592 #
ED.PACK.10               34595   34602 #
ED.PACK.20               34605 #
ED.POD                   33561 # 33572
ED.POD.10                33563   33607 #
ED.POD.15                33621 # 34759
ED.POD.16                33627 # 33960
ED.POD.20                33611   33635 #
ED.POD.30                33639   33646 #
ED.POD.40                33650   33657 #
ED.READ                  34017   34141   34249   34347 #
ED.READ.10               34355   34366 #
ED.READ.50               34363   34403 #
ED.READ.DONE             34373   34391 #
ED.READ.FETCH.BYTE       34369   34376 #
ED.READ.FETCH.OK         34380   34385 #
ED.REPLACE.SIGN          33810   33940 #
ED.RESV.OPER             33619   33654   33660   34133   34241   34359   34539   34597 #
ED.SAVE.NEW.SIGN         33820   33833 #
ED.SIGN.FILL             33542   33547 #
ED.STORE                 33899   34068   34085   34193   34198   34310   34315   34319   34430 #
ED.STORE.DONE            34437   34444 #
ED.STORE.NOINT           33732   33739   33859   33863   34487 # 34741
ED.STORE.NOINT.10        34492 #
ED.STORE.NOINT.DONE      33953   34495   34507 #
ED.STORE.PACK            34441   34460 #
ED.UNPACK                34652 # 34772   34772   34772   34772
ED.UNPACK.03             34690 # 34697
ED.UNPACK.05             34693   34699 #
FI.ADDD2.100             16762   16789 # 16845
FI.ADDD2.20              16764 # 16787
FI.ADDD2.40              16771   16774 # 16812   16827   16957
FI.ADDD2.MEM             16784 # 17663   17663   17663   17825   17825   17825   17898   17898   17898   17943   17943
                         17943
FI.ADDD2.REG             16759 # 17662   17662   17824   17824   17897   17897   17942   17942
FI.ADDD3.20              16820 # 17715   17715   17715   17784   17784   17784   17811   17811   17811
```

```
FI.POLYF.70                17034   17037 # 17056
FI.POLYF.FPD.10            17153 # 17160   39056
FI.POLYF.FPD.RES           17193 # 17928   17928
FI.POLYF.FPD.SAVE          39051 #
FI.POLYF.OFF               17114 # 17200
FI.POLYFD.10               16987   17068 # 17925   17925   17925
FI.POLYFD.OFF              17107   17115 #
FI.POLYFD.RES.10           16280 # 17196   17211
FI.POLYFD.RW1              17078 # 17261   17266
FI.POLYFD.RW2              17083 # 17270
FI.POLYFD.RW3              17087 #
FI.POLYFD.RW3.5            17010   17016 # 17091   17286
FI.POLYFD.RW4              17020   17041   17093 #
FI.POLYFD.SET              17026   17104 # 17275
FP.ACBD                    15469 # 17648   17648   17648   17648   17648   17648
FP.ACBD.100                13446 # 17649   17649   17649   17649   17649   17649
FP.ACBD.400                15496   15500 #
FP.ACBD.450                15505   15511   15513 # 15527   15532   15538   15544
FP.ACBF                    15297 # 17656   17656   17656   17656   17656   17656
FP.ACBF.400                15322   15326 #
FP.ACBF.450                15330   15335   15338 # 15516
FP.ACBF.500                15315   15346 # 15482
FP.ACBF.525                15358   15364 #
FP.ACBF.550                15353   15369 #
FP.ACBF.570                15367   15372   15380   15383 #
FP.ACBF.580                15386   15404 #
FP.ADD.SIGN2               12706   12712   12726   12731   12736 #
FP.ADD.SIGNOP1             12338   12380   12697 #
FP.ADD.SIGNOP2             12406   12719 # 13611
FP.ADDD.20                 13458 # 17663   17663   17663   17670   17670   17670   17943   17943   17943   17950   17950
                           17950
FP.ADDD.25                 13464 # 16673
FP.ADDD.30                 13476 #
FP.ADDD.SIGNOP1            12691 # 13573   13578
FP.ADDD2                   13149 # 17662   17662   17662
FP.ADDD3                   13225 # 17669   17669   17669
FP.ADDDOP1                 11552   11641   13153   13229   13307 # 13450
FP.ADDDOP1.20              13314   13325   13328 # 13951   13960
FP.ADDDOP1.30              13333   13340 #
FP.ADDDOP2                 13350 # 13462   16656
FP.ADDDOP2.20              13357   13369 #
FP.ADDDRTN                 13469   13569 #
FP.ADDDRTN.120             13582   13607 #
FP.ADDDRTN.125             13630   13638 #
FP.ADDDRTN.130             13636   13643 #
FP.ADDDRTN.200             13592   13649 #
FP.ADDDRTN.250             13603   13655 #
FP.ADDDRTN.300             13646   13653   13661 #
FP.ADDDRTN.320             13668 #
FP.ADDDRTN.620             13684   13696 #
FP.ADDDRTN.625             13699   13704 #
FP.ADDDRTN.RNGCHK          13588   13625   13680 #
FP.ADDF.2                  11980 # 12058   12063   12070
FP.ADDF.SUB                12157   12167   12219 #
FP.ADDF23                  11975 # 17677   17677   17677   17684   17684   17684
```

Cross Reference Listing - Field Names and Defined Values

```
FP.ADDFOP1              11455   11978   12125 # 15305
FP.ADDFOP2              11984   12149 # 15310
FP.ADDRTN               12224   12329 # 15889
FP.ADDRTN.100           12333   12376 # 12385
FP.ADDRTN.20            12356 # 12390   12411
FP.ADDRTN.200           12343   12393 # 12400
FP.ADDRTN.250           12396   12402 #
FP.ADDRTN.30            12354   12363 #
FP.CMPD                 11090 # 17694   17694   17694
FP.CMPD.10              11103   11107   11112 #
FP.CMPD.20              11119 # 17695   17695   17695
FP.CMPD.30              11141   11143 #
FP.CMPD.40              11132   11152 #
FP.CMPD.50              11146   11161 #
FP.CMPD.60              11158   11172 #
FP.CMPF                 10974 # 17701   17701   17701
FP.CMPF.10              10978   10986 #
FP.CMPF.30              10983   11005 #
FP.CMPF.40              11021   11024 #
FP.CMPF.50              11012   11032 #
FP.CVT.BWL.FD           11317 # 17714   17714   17714   17721   17721   17721   17783   17783   17783   17810   17810
                        17810   17817   17817   17817
FP.CVT.BWL.FD.100       11349 # 17715   17715   17715   17722   17722   17722   17784   17784   17784   17811   17811
                        17811   17818   17818   17818
FP.CVT.BWL.FD.200       11337   11353   11356 # 11369
FP.CVT.BWL.FD.220       11359   11371 #
FP.CVT.BWL.FD.250       11377 # 11387
FP.CVT.BWL.FD.300       11324 # 11345
FP.CVTD.150             11568 # 17729   17729   17729   17743   17743   17743   17750   17750   17750   17798   17798
                        17798
FP.CVTD.BWL             11548 # 17728   17728   17728   17742   17742   17742   17749   17749   17749
FP.CVTDF                11754 # 17736   17736   17736
FP.CVTDF.100            11771   11782   11785 # 11825
FP.CVTF.100             11441   11452 # 11630
FP.CVTF.200             11443 # 11473
FP.CVTF.250             11448   11462 # 11564
FP.CVTF.300             11467   11476 #
FP.CVTF.400             11491 # 11672   11678
FP.CVTF.450             11482   11496 #
FP.CVTF.BWL             11438 # 17756   17756   17756   17770   17770   17770   17777   17777   17777
FP.CVTFD                11221 # 17762   17762   17762
FP.CVTFD.20             11235   11240   11245   11248 #
FP.CVTFD.30             11256 # 17763   17763   17763
FP.CVTFI.30             15110   15134 # 15141
FP.CVTFI.35             15115   15149 #
FP.CVTFI.45             15163   15173 # 15216
FP.CVTFI.50             15132   15185 #
FP.CVTFI.55             15188   15210 #
FP.CVTFI.70             15207   15218 #
FP.CVTFI.75             15153   15222 #
FP.CVTFI.90             15079   15095   15235 #
FP.CVTFI.ENT            15084   15100 #
FP.CVTFI.SUB            11459   11571   15085 #
FP.CVTFI.SUBALT         14702   15091 #
FP.CVTLF                11332 # 17791   17791   17791
```

Cross Reference Listing - Field Names and Defined Values

```
FP.MULD.SUB.10          14216 # 14220
FP.MULD.SUB.20          14231 # 14235
FP.MULD.SUB.30          14251 # 14255
FP.MULD2                13759 # 17897   17897   17897
FP.MULD3                13778 # 17904   17904   17904
FP.MULDOP1              13763   13782   13944 # 14306   14325   14937   16041   16557
FP.MULDOP2              13863   13963 # 14420   14961   16248   16422
FP.MULDOP2.20           13970   13979 #
FP.MULF.10              12829   12836 #
FP.MULF.100             12847 # 12852
FP.MULF.200             12846   12874 #
FP.MULF.250             12878   12885 #
FP.MULF23               12831 # 17912   17912   17912   17918   17918   17918
FP.MULFOP1              12834   12947 # 13053   14641   15685   15771
FP.MULFOP2              12840   12972 # 13058   14657   15726   15755   15789   16027
FP.NORM.ADD.ALKC        12550   12599   12624 #
FP.NORM.DBL             12529 # 13480
FP.NORM.DBL.10          12534   12583 #
FP.NORM.DBL.11          12601 # 12621
FP.NORM.DBL.50          12587   12609 #
FP.NORM.MUDI            12508 # 12866   13091   13885   14437   16668
FP.NORM.SNG             12238   12513   12519   12522 # 14713   15905
FP.NORM.SNG.10          12527   12539 #
FP.NORM.SNG.20          12552 # 12607
FP.PACK.T2T5            11329   11878 #
FP.PCDELTA              16008 #
FP.POLYD                16412 # 17924   17924   17924
FP.POLYD.10             16433 # 17925   17925   17925
FP.POLYD.100            16611   16618   16635 #
FP.POLYD.20             16270   16470 # 16506   16512
FP.POLYD.22             16474   16480 #
FP.POLYD.220            16478   16633   16646 #
FP.POLYD.25             16462   16468   16508 #
FP.POLYD.30             16518 # 16650
FP.POLYD.35             16274   16530 #
FP.POLYD.40             16449   16536 #
FP.POLYD.60             16502   16546 #
FP.POLYD.65             16553 # 16630
FP.POLYD.72             16576 # 16599   16604
FP.POLYD.75             16584   16589   16606 # 16624
FP.POLYD.80             16570   16595   16620 #
FP.POLYD.90             16551   16626 #
FP.POLYD.FIX            16643   16652 #
FP.POLYD.FIX2           16661   16664 #
FP.POLYD.FPD.RES        16223 # 17921   17921
FP.POLYD.FPD.SAV.10     16134 # 39008
FP.POLYD70.FIX          16493   16586 #
FP.POLYF                15682 # 17932   17932   17932
FP.POLYF.10             15680   15687 #
FP.POLYF.100            15830   15896 #
FP.POLYF.112            15907 #
FP.POLYF.115            15699   15894   15914 #
FP.POLYF.20             15704 # 15747   15923
FP.POLYF.22             15716 #
FP.POLYF.25             15722 # 16219
```

```
FP.POLYF.50              15734 # 15919   16214
FP.POLYF.55              15731   15744 #
FP.POLYF.60              15737   15750 #
FP.POLYF.62              15760   15768 #
FP.POLYF.63              15766   15773 #
FP.POLYF.65              15779 # 15783
FP.POLYF.68              15801   15809 # 15852
FP.POLYF.70              15807   15814 # 15857   15862   15870
FP.POLYF.75              15777   15833 #
FP.POLYF.85              15819   15872 #
FP.POLYF.90              15876   15884 #
FP.POLYF.FIX25.00        15794   15844 #
FP.POLYF.FIX25.10        15847   15866 #
FP.POLYF.FPD.05          16205   16209 #
FP.POLYF.FPD.RES         16196 # 17928   17928
FP.POLYF.FPD.SAV.10      16111 # 39003
FP.POLYFD.25             16037 # 16235
FP.POLYFD.BT30           15711   16067 # 16525
FP.POLYFD.BT30A          16071 #
FP.POLYFD.END            15707   16045 # 16278   16521
FP.POLYFD.FPD.10         16114   16116 # 16137
FP.POLYFD.FPD.50         16120   16122 # 16141
FP.POLYFD.RD             15992   16017 # 16534
FP.POLYFD.RES.08         16251   16253 #
FP.POLYFD.RES.10         16200   16256   16281 #
FP.POLYFD.SUBS           15690   15983 # 16442
FP.POLYFD.VAR3           15695   16050 # 16453
FP.PREOPS.POLYD          14035 # 16488
FP.PREOPS.POLYD10        14039   14044   14046 #
FP.PREOPS.QT2            14049   14059   14070   14095 #
FP.PREOPS.QT3            14091 # 14949
FP.PREOPS.SUB            13878   14067 #
FP.PREOPS.SUBALT         14053   14063   14074 # 14987
FP.QT1.SC17.CHKINT       14105 # 14214   14249
FP.QT6.SC17              14100 # 14229
FP.SC17                  13872   14103   14110 # 14451   16483
FP.SETVIRD1              11495 # 14809   14822
FP.SUBD2                 13144 # 17942   17942   17942
FP.SUBD3                 13220 # 17949   17949   17949
FP.SUBF.100              12049 #
FP.SUBF2                 12031 # 17956   17956   17956
FP.SUBF23.20             12044 # 17957   17957   17957   17964   17964   17964
FP.SUBF3                 12038 # 17963   17963   17963
FP.T3TOT2                12616   15180 #
FP.T3ZERO                12591   15137   15144 #
FP.TESTPSL               12566   12570   12631 # 14740
FP.TSTD                  10912 # 17969   17969   17969   17969   17969   17969
FP.TSTF                  10907 # 17976   17976   17976   17976   17976   17976
FP.TSTF.10               10910   10915   10920 #
FP.VALIDCC               15395   15401   15412 #
FP.WRITE.FIRST           11364   11882   11890   11905 #
FP.WRITE.SEC             10754   10767   10777   11262   11917   11920 # 13169   13248   13774   13800   14317   14343
FP.WRITE23               11894 # 11988   12870   12883   13075   13095
FP.ZEROTEMP1             13362   15229 #
FP.ZEROTEMP2             12538 # 15192   15202
```

```
FP.ZEROTEMP3            14646   15177   15183   15195 # 15226
FREE.0458               9433 #
FREE.0459               9438 #
FREE.045A               9422 #
FREE.045B               9428 #
FREE.045C               9457 #
FREE.045D               9462 #
FREE.045E               9468 #
FREE.04C7               44802 #
FREE.0639               15797 #
FREE.063B               15803 #
FREE.0659               16565 #
FREE.065B               16572 #
FREE.0707               16464 #
FREE.071D               17632 #
FREE.07C0               47933 #
FREE.07C1               47940 #
FREE.0D30               22089 #
FREE.0D6E               29667 #
FREE.0D6F               29672 #
FREE.0E82               36099 #
FREE.0F83               36104 #
FREE.0E8D               35505 #
FREE.0E92               35605 #
FREE.0EF7               38376 #
FREE.0F8C               39460 #
FREE.0F8D               39466 #
FREE.0FAE               38360 #
FREE.0FB2               38447 #
FREE.1672               42563 #
FREE.1769               39794 #
FREE.176B               39825 #
FX.ACBD.400             15499 #
FX.ACBG.345             15484 #
FX.ACBG.350             15491 #
FX.ACBG.500             15345 #
FX.ACBH.500             15523 #
FX.ADDDRTN              13568 #
FX.ADDFOP1              12124 #
FX.ADDHOP1              13306 #
FX.ADJTMPS.LFT          47830 # 48049
FX.ADJTMPS.LFT2         47826 # 47960
FX.ADJTMPS.RT3          47879 #
FX.ADJTMPS.RT4          47875 #
FX.ADJTMPS.RT5          47871 #
FX.CCS                  47464 #
FX.CVTFI.75             15223 #
FX.CVTFI.90             15236 #
FX.CVTFI.CHKOVR         47975   48004 #
FX.CVTFI.CHKOVR.10      48014   48025 #
FX.CVTFI.CHKOVR.20      48008   48018   48022   48029   48038   48046 # 48053
FX.CVTFI.HSUB.10        47943   47945 #
FX.CVTFI.HSUB.100       47970   47986 #
FX.CVTFI.HSUB.110       47937   47995 #
FX.CVTFI.HSUB.20        47950 #
```

```
FX.CVTFI.HSUB.30        47956 # 47984
FX.CVTFI.HSUB.50        47954   47967 #
FX.CVTFI.SUBG           15073   15075 #
FX.CVTFI.SUBG3          15081 #
FX.CVTFI.SUBGC          15069 #
FX.CVTS.CC              47537 #
FX.DIVD.50              14444 #
FX.DSIGNSET             16056 #
FX.EMODG.100            14754 #
FX.EMODG.300            14862 #
FX.EMODH.FPD            39109 #
FX.EMODH.INTWT          14760 #
FX.GOPER.POLY.ARG       47693 #
FX.GOPER1.10            47611 #
FX.GOPER1.20            47615   47624 #
FX.GOPER1.30            47661   47664 #
FX.GOPER12              47631   47641   47657 # 47681   47691
FX.GOPER2.10            47617 #
FX.GOPER2.20            47621   47674 # 47697
FX.HIDBIT               47649 # 47667   47671
FX.IE.FD.OPCOD          39203 # 39514   39514   39514   39514
FX.IE.FD.OPCOD2         39210   39236 #
FX.IE.FD.OPCOD3         39239   39242 #
FX.IE.FD.OPCOD4         39213 # 39245
FX.IE.FD.RET            22214   22214   22214   22214   22214   22214   22215   22215   22215   22215   22215   22215
                        39231 # 39516   39516   39516   39516   39516   39516   39517   39517   39517   39517   39517
                        39517
FX.IE.FPD.RES           39261 # 39513   39513   39513   39513
FX.IE.FPD.RES2          39269   39285 #
FX.IE.FPD.RES3          39288   39291 #
FX.IE.FPD.RES4          39272 # 39295
FX.IE.WCS.ENABLED       39217   39220 #
FX.INCVA                15546 #
FX.INT.OVR              47564 #
FX.INT.OVR.TST          47541   47546   47551   47554 # 47575
FX.LFTSHFT3             47842 # 47992
FX.LFTSHFT4             47846 #
FX.MNEGGH.10            10861 #
FX.MNEGGH.20            10870 #
FX.MT2Q                 10998 #
FX.MULSIGN              47491 #
FX.NORM.ADD.ALKC        12625 # 47745
FX.NORM.GFL             47717 #
FX.NORM.GFL.10          47722   47729 #
FX.NORM.GFL.20          47747 # 47782
FX.NORM.GFL.30          47752   47755 #
FX.NORM.GFL.50          47733   47770 #
FX.PACK.GFL             47786 # 47798
FX.PACK.GFL.10          47791 # 47805
FX.PACK.GFL.12          47795 #
FX.PACK.GFL.15          47759   47802 #
FX.PACK.GFL.30          47789   47812 #
FX.PLOSL15              47897 #
FX.PL4                  47517 #
FX.POLYD.220            16647 #
```

```
FX.POLYFD.BT30          16066 #
FX.POLYFD.RES.10        16282 #
FX.POLYG.FPD.SAV.10     16133 # 39099
FX.POLYG.FPD.SAVE       39094 #
FX.POLYG.SUB            15994 #
FX.POLYG.VAR3           16049 #
FX.POLYGH.45            16541 #
FX.POLYH.FPD.SAVE       39101 #
FX.PREOPS.POLYGT9       14056 #
FX.PREOPS.QT3           14090 #
FX.PREOPS.SUB           14066 #
FX.PREOPS.SUBALTG       14080 #
FX.PROP.CRY             47854 #
FX.ROTIMP.T5T3          47884 #
FX.SL11P4               47513 #
FX.SL15                 47901 #
FX.T3GETST4             13637 #
FX.T3TOT2               15179 # 47777
FX.T3ZERO               15143 # 47737
FX.T5ZERO               47837 #
FX.TESTPSL              12630 # 47763   47767
FX.TSTGH.10             10919 #
FX.TSTV.PSL             47568 #
FX.TYPE.HID             47644 #
FX.VALIDCC              15411 #
FX.WRITE.SEC            11919 # 47470   47474   47478
FX.WRITE.SEC.10         11926 #
FX.ZERORT12             47906 #
FX.ZEROTEMP1            13956   15228 # 47686
FX.ZEROTEMP2            12537 #
FX.ZEROTEMP3            15194 # 47636
GL.DBG.NOP              37890 # 37892
GL.NOP.IRD1             20914   21390   21400   24318   24372   30002   36527   37885 # 38106
IE.105                  36354   37339 # 37346
IE.110                  37341   37348 #
IE.20                   37213   37225 #
IE.25                   37228   37234 #
IE.50                   37269 # 38298
IE.60                   37242   37273 #
IE.65                   37271   37279   37281 #
IE.70                   37299   37309 #
IE.75                   37303   37314 #
IE.80                   37295   37319 #
IE.90                   37307   37312   37317   37322   37324 #
IE.ABORT                37178 # 38655   38722
IE.ACV                  21229   38394   38400   38402 # 40702   40763   40822   40863
IE.ACV.BUT              38412 #
IE.ACV.BUT.1            38414   38422 #
IE.ACV.CHK.CONSOLE      38374   38397 # 39896   39984
IE.ACV.PUSH.PARAM       38480 # 39167
IE.ACV.RBACK            38419   38431 # 38438   40414   40432
IE.ACV.READ             38349 #
IE.ACV.SAV.FA           38428   38490 #
IE.ACV.WRITE            38343 #
IE.ACV.XB               38382 #
```

Cross Reference Listing - Field Names and Defined Values

Cross Reference Listing - Field Names and Defined Values

39571	39572	39572	39572	39572	39572	39572	43272	43272	43272	43272	43273
43273	43275	43275	43275	43275	43276	43276	43278	43278	43281	43281	43284
43284	43285	43285	43287	43287	43288	43288	43290	43290	43291	43291	43293
43293	43294	43294	43296	43296	43296	43296	43297	43297	43299	43299	43299
43299	43300	43300	43302	43302	43302	43302	43303	43303	43305	43305	43305
43305	43306	43306	43308	43308	43308	43308	43309	43309	43309	43309	43311
43311	43311	43311	43312	43312	43312	43312	43314	43314	43314	43314	43315
43315	43315	43315	43317	43317	43317	43317	43318	43318	43318	43318	43321
43321	43321	43321	43322	43322	43322	43322	43324	43324	43324	43324	43325
43325	43325	43325	43327	43327	43330	43330	43333	43333	43336	43336	43339
43339	43342	43342	43345	43345	43348	43348	43351	43351	43354	43354	43357
43357	43360	43360	43363	43363	43363	43363	43364	43364	43364	43364	43366
43366	43366	43366	43367	43367	43367	43367	43369	43369	43369	43369	43370
43370	43370	43370	43372	43372	43372	43372	43373	43373	43373	43373	43376
43376	43376	43376	43377	43377	43377	43377	43379	43379	43379	43379	43380
43380	43380	43380	43382	43382	43382	43382	43383	43383	43383	43383	43385
43385	43385	43385	43386	43386	43386	43386	43388	43388	43388	43388	43389
43389	43389	43389	43391	43391	43391	43391	43392	43392	43392	43392	43394
43394	43394	43394	43395	43395	43395	43395	43397	43397	43397	43397	43398
43398	43398	43398	43400	43400	43400	43400	43401	43401	43403	43403	43403
43403	43404	43404	43406	43406	43409	43409	43412	43412	43415	43415	43418
43418	43418	43418	43419	43419	43421	43421	43421	43421	43422	43422	43424
43424	43424	43424	43425	43425	43427	43427	43427	43427	43428	43428	43431
43431	43432	43432	43434	43434	43435	43435	43437	43437	43437	43437	43438
43438	43438	43438	43440	43440	43440	43440	43441	43441	43441	43441	43443
43443	43443	43443	43444	43444	43444	43444	43446	43446	43446	43446	43447
43447	43447	43447	43449	43449	43449	43449	43450	43450	43452	43452	43452
43452	43453	43453	43455	43455	43455	43455	43456	43456	43456	43456	43458
43458	43458	43458	43459	43459	43461	43461	43461	43461	43462	43462	43464
43464	43464	43464	43465	43465	43467	43467	43467	43467	43468	43468	43470
43470	43471	43471	43473	43473	43474	43474	43476	43476	43477	43477	43479
43479	43480	43480	43482	43482	43483	43483	43486	43486	43487	43487	43489
43489	43490	43490	43492	43492	43493	43493	43495	43495	43495	43495	43496
43496	43498	43498	43498	43498	43499	43499	43501	43501	43501	43501	43502
43502	43504	43504	43504	43504	43505	43505	43507	43507	43510	43510	43513
43513	43516	43516	43519	43519	43520	43520	43522	43522	43523	43523	43525
43525	43525	43525	43526	43526	43528	43528	43528	43528	43529	43529	43531
43531	43531	43531	43532	43532	43532	43532	43534	43534	43534	43534	43535
43535	43535	43535	43537	43537	43537	43537	43538	43538	43538	43538	43541
43541	43541	43541	43542	43542	43542	43542	43544	43544	43544	43544	43545
43545	43545	43545	43547	43547	43547	43547	43548	43548	43548	43548	43550
43550	43550	43550	43551	43551	43551	43551	43553	43553	43553	43553	43554
43554	43554	43554	43556	43556	43556	43556	43557	43557	43557	43557	43559
43559	43559	43559	43560	43560	43560	43560	43562	43562	43562	43562	43563
43563	43563	43563	43565	43565	43565	43565	43566	43566	43566	43566	43568
43568	43568	43568	43569	43569	43569	43569	43571	43571	43571	43571	43572
43572	43572	43572	43574	43574	43574	43574	43575	43575	43575	43575	43577
43577	43577	43577	43578	43578	43578	43578	43580	43580	43580	43580	43581
43581	43581	43581	43583	43583	43583	43583	43584	43584	43584	43584	43586
43586	43586	43586	43587	43587	43587	43587	43589	43589	43589	43589	43590
43590	43590	43590	43592	43592	43592	43592	43593	43593	43593	43593	43596
43596	43596	43596	43597	43597	43597	43597	43599	43599	43599	43599	43600
43600	43600	43600	43602	43602	43602	43602	43603	43603	43603	43603	43605
43605	43605	43605	43606	43606	43606	43606	43608	43608	43608	43608	43609
43609	43609	43609	43611	43611	43611	43611	43612	43612	43612	43612	43614

43614	43614	43614	43615	43615	43615	43615	43617	43617	43617	43617	43618
43618	43618	43618	43620	43620	43620	43620	43621	43621	43621	43621	43623
43623	43623	43623	43624	43624	43624	43624	43626	43626	43626	43626	43627
43627	43627	43627	43629	43629	43629	43629	43630	43630	43630	43630	43632
43632	43632	43632	43633	43633	43633	43633	43635	43635	43635	43635	43636
43636	43636	43636	43638	43638	43638	43638	43639	43639	43639	43639	43641
43641	43641	43641	43642	43642	43642	43642	43644	43644	43644	43644	43645
43645	43645	43645	43647	43647	43647	43647	43648	43648	43648	43648	43651
43651	43651	43651	43652	43652	43652	43652	43654	43654	43654	43654	43655
43655	43655	43655	43657	43657	43657	43657	43658	43658	43658	43658	43660
43660	43660	43660	43661	43661	43661	43661	43663	43663	43663	43663	43664
43664	43664	43664	43666	43666	43666	43666	43667	43667	43667	43667	43669
43669	43669	43669	43670	43670	43670	43670	43672	43672	43672	43672	43673
43673	43673	43673	43675	43675	43675	43675	43676	43676	43676	43676	43678
43678	43678	43678	43679	43679	43679	43679	43681	43681	43681	43681	43682
43682	43682	43682	43684	43684	43684	43684	43685	43685	43685	43685	43687
43687	43687	43687	43688	43688	43688	43688	43690	43690	43690	43690	43691
43691	43691	43691	43693	43693	43693	43693	43694	43694	43694	43694	43696
43696	43696	43696	43697	43697	43697	43697	43699	43699	43699	43699	43700
43700	43700	43700	43702	43702	43702	43702	43703	43703	43703	43703	43706
43706	43706	43706	43707	43707	43707	43707	43709	43709	43709	43709	43710
43710	43710	43710	43712	43712	43712	43712	43713	43713	43713	43713	43715
43715	43715	43715	43716	43716	43716	43716	43718	43718	43718	43718	43719
43719	43719	43719	43721	43721	43721	43721	43722	43722	43722	43722	43724
43724	43724	43724	43725	43725	43725	43725	43727	43727	43727	43727	43728
43728	43728	43728	43730	43730	43730	43730	43731	43731	43731	43731	43733
43733	43733	43733	43734	43734	43734	43734	43736	43736	43736	43736	43737
43737	43737	43737	43739	43739	43739	43739	43740	43740	43740	43740	43742
43742	43742	43742	43743	43743	43743	43743	43745	43745	43745	43745	43746
43746	43746	43746	43748	43748	43748	43748	43749	43749	43749	43749	43751
43751	43751	43751	43752	43752	43752	43752	43754	43754	43754	43754	43755
43755	43755	43755	43757	43757	43757	43757	43758	43758	43758	43758	43761
43761	43761	43761	43762	43762	43762	43762	43764	43764	43764	43764	43765
43765	43765	43765	43767	43767	43767	43767	43768	43768	43768	43768	43770
43770	43770	43770	43771	43771	43771	43771	43773	43773	43773	43773	43774
43774	43774	43774	43776	43776	43776	43776	43777	43777	43777	43777	43779
43779	43779	43779	43780	43780	43780	43780	43782	43782	43782	43782	43783
43783	43783	43783	43785	43785	43785	43785	43786	43786	43786	43786	43788
43788	43788	43788	43789	43789	43789	43789	43791	43791	43791	43791	43792
43792	43792	43792	43794	43794	43794	43794	43795	43795	43795	43795	43797
43797	43797	43797	43798	43798	43798	43798	43800	43800	43800	43800	43801
43801	43801	43801	43803	43803	43803	43803	43804	43804	43804	43804	43806
43806	43806	43806	43807	43807	43807	43807	43809	43809	43809	43809	43810
43810	43810	43810	43812	43812	43812	43812	43813	43813	43813	43813	43816
43816	43816	43816	43817	43817	43817	43817	43819	43819	43819	43819	43820
43820	43820	43820	43822	43822	43822	43822	43823	43823	43823	43823	43825
43825	43825	43825	43826	43826	43826	43826	43828	43828	43828	43828	43829
43829	43829	43829	43831	43831	43831	43831	43832	43832	43832	43832	43834
43834	43834	43834	43835	43835	43835	43835	43837	43837	43837	43837	43838
43838	43838	43838	43840	43840	43840	43840	43841	43841	43841	43841	43843
43843	43843	43843	43844	43844	43844	43844	43846	43846	43846	43846	43847
43847	43847	43847	43849	43849	43849	43849	43850	43850	43850	43850	43852
43852	43852	43852	43853	43853	43853	43853	43855	43855	43855	43855	43856
43856	43856	43856	43858	43858	43858	43858	43859	43859	43859	43859	43861
43861	43861	43861	43862	43862	43862	43862	43864	43864	43864	43864	43865

Cross Reference Listing - Field Names and Defined Values

```
                                43865  43865  43865  43867  43867  43867  43867  43868  43868  43868  43868  43871
                                43871  43871  43871  43872  43872  43872  43872  43874  43874  43874  43874  43875
                                43875  43875  43875  43877  43877  43877  43877  43878  43878  43878  43878  43880
                                43880  43880  43880  43881  43881  43881  43881  43883  43883  43883  43883  43884
                                43884  43884  43884  43886  43886  43886  43886  43887  43887  43887  43887  43889
                                43889  43889  43889  43890  43890  43890  43890  43892  43892  43892  43892  43893
                                43893  43893  43893  43895  43895  43895  43895  43896  43896  43896  43896  43898
                                43898  43898  43898  43899  43899  43899  43899  43901  43901  43901  43901  43902
                                43902  43902  43902  43904  43904  43904  43904  43905  43905  43905  43905  43907
                                43907  43907  43907  43908  43908  43908  43908  43910  43910  43910  43910  43911
                                43911  43911  43911  43913  43913  43913  43913  43914  43914  43914  43914  43916
                                43916  43916  43916  43917  43917  43917  43917  43919  43919  43919  43919  43920
                                43920  43920  43920  43922  43922  43922  43922  43923  43923  43923  43923  43926
                                43926  43926  43926  43927  43927  43927  43927  43929  43929  43929  43929  43930
                                43930  43930  43930  43932  43932  43932  43932  43933  43933  43933  43933  43935
                                43935  43935  43935  43936  43936  43936  43936  43938  43938  43938  43938  43939
                                43939  43939  43939  43941  43941  43941  43941  43942  43942  43942  43942  43944
                                43944  43944  43944  43945  43945  43945  43945  43947  43947  43947  43947  43948
                                43948  43948  43948  43950  43950  43950  43950  43951  43951  43951  43951  43953
                                43953  43953  43953  43954  43954  43954  43954  43956  43956  43956  43956  43957
                                43957  43957  43957  43959  43959  43959  43959  43960  43960  43960  43960  43962
                                43962  43962  43962  43963  43963  43963  43963  43965  43965  43965  43965  43966
                                43966  43966  43966  43968  43968  43968  43968  43969  43969  43969  43969  43971
                                43971  43971  43971  43972  43972  43972  43972  43974  43974  43974  43974  43975
                                43975  43977  43977  43977  43977  43978  43978  43981  43981  43981  43981  43982
                                43982  43984  43984  43984  43984  43985  43985  43987  43987  43987  43987  43988
                                43988  43988  43988  43990  43990  43990  43990  43991  43991  43991  43991  43993
                                43993  43993  43993  43994  43994  43994  43994  43996  43996  43996  43996  43997
                                43997  43999  43999  43999  43999  44000  44000  44002  44002  44002  44002  44003
                                44003  44003  44003  44005  44005  44005  44005  44006  44006  44006  44006  44008
                                44008  44011  44011  44014  44014  44014  44014  44015  44015  44017  44017  44017
                                44017  44018  44018  44020  44020  44020  44020  44021  44021  44023  44023  44023
                                44023  44024  44024  44026  44026  44026  44026  44027  44027  44029  44029  44029
                                44029  44030  44030  44030  44030  44032  44032  44032  44032  44033  44033  44033
                                44033  44036  44036  44036  44036  44037  44037  44037  44037  44039  44039  44039
                                44039  44040  44040  44040  44040  44042  44042  44042  44042  44043  44043  44045
                                44045  44045  44045  44046  44046
IE.CACHE.PARITY                 37122  #
IE.CHECK.DBL.ERR                37386  #  37563
IE.CHECK.FOR.CACHE              37663     37688  #
IE.CHECK.FOR.CACHE.CONTINUE               37832  #
IE.CLEAR.INT                    37410     37417  #
IE.CLEAR.TRP                    38541     38547     38554  #
IE.CLR.GPRS                     39485     39491  #
IE.CLR.GPRS15                   39495  #  39508
IE.CM.BPT                       38612  #  43382     43382  43383  43383
IE.CM.EMT                       38624  #  43437     43437  43438  43438  43440  43440  43441  43441  43443  43443  43444
                                43444     43446     43446  43447  43447
IE.CM.EXCEP                     38644     38651  #
IE.CM.FAULT                     38610     38627     38632  38638  38647  #  42711
IE.CM.ILLINS                    38635  #  42415     42460
IE.CM.IOT                       38618  #  43455     43455  43456  43456
IE.CM.ODD                       38641  #  39489     42407  42574  42584  42607  42686  42698  42824  42834  42850  42970
                                42980     42996     43112  43121  43135  43143  43172  43219  43246  43255  43264
IE.CM.RESOP                     38607  #  42703     43531  43531  43532  43532  43534  43534  43535  43535  43537  43537
```

43538	43538	43541	43541	43542	43542	43544	43544	43545	43545	43547	43547
43548	43548	43550	43550	43551	43551	43553	43553	43554	43554	43556	43556
43557	43557	43559	43559	43560	43560	43562	43562	43563	43563	43565	43565
43566	43566	43568	43568	43569	43569	43571	43571	43572	43572	43574	43574
43575	43575	43577	43577	43578	43578	43580	43580	43581	43581	43583	43583
43584	43584	43586	43586	43587	43587	43589	43589	43590	43590	43592	43592
43593	43593	43596	43596	43597	43597	43599	43599	43600	43600	43602	43602
43603	43603	43605	43605	43606	43606	43608	43608	43609	43609	43611	43611
43612	43612	43614	43614	43615	43615	43617	43617	43618	43618	43620	43620
43621	43621	43623	43623	43624	43624	43626	43626	43627	43627	43629	43629
43630	43630	43632	43632	43633	43633	43635	43635	43636	43636	43638	43638
43639	43639	43641	43641	43642	43642	43644	43644	43645	43645	43647	43647
43648	43648	43651	43651	43652	43652	43654	43654	43655	43655	43657	43657
43658	43658	43660	43660	43661	43661	43663	43663	43664	43664	43666	43666
43667	43667	43669	43669	43670	43670	43672	43672	43673	43673	43675	43675
43676	43676	43678	43678	43679	43679	43681	43681	43682	43682	43684	43684
43685	43685	43687	43687	43688	43688	43690	43690	43691	43691	43693	43693
43694	43694	43696	43696	43697	43697	43699	43699	43700	43700	43702	43702
43703	43703	43706	43706	43707	43707	43709	43709	43710	43710	43712	43712
43713	43713	43715	43715	43716	43716	43718	43718	43719	43719	43721	43721
43722	43722	43724	43724	43725	43725	43727	43727	43728	43728	43730	43730
43731	43731	43733	43733	43734	43734	43736	43736	43737	43737	43739	43739
43740	43740	43742	43742	43743	43743	43745	43745	43746	43746	43748	43748
43749	43749	43751	43751	43752	43752	43754	43754	43755	43755	43757	43757
43758	43758	43761	43761	43762	43762	43764	43764	43765	43765	43767	43767
43768	43768	43770	43770	43771	43771	43773	43773	43774	43774	43776	43776
43777	43777	43779	43779	43780	43780	43782	43782	43783	43783	43785	43785
43786	43786	43788	43788	43789	43789	43791	43791	43792	43792	43794	43794
43795	43795	43797	43797	43798	43798	43800	43800	43801	43801	43803	43803
43804	43804	43806	43806	43807	43807	43809	43809	43810	43810	43812	43812
43813	43813	43816	43816	43817	43817	43819	43819	43820	43820	43822	43822
43823	43823	43825	43825	43826	43826	43828	43828	43829	43829	43831	43831
43832	43832	43834	43834	43835	43835	43837	43837	43838	43838	43840	43840
43841	43841	43843	43843	43844	43844	43846	43846	43847	43847	43849	43849
43850	43850	43852	43852	43853	43853	43855	43855	43856	43856	43858	43858
43859	43859	43861	43861	43862	43862	43864	43864	43865	43865	43867	43867
43868	43868	43871	43871	43872	43872	43874	43874	43875	43875	43877	43877
43878	43878	43880	43880	43881	43881	43883	43883	43884	43884	43886	43886
43887	43887	43889	43889	43890	43890	43892	43892	43893	43893	43895	43895
43896	43896	43898	43898	43899	43899	43901	43901	43902	43902	43904	43904
43905	43905	43907	43907	43908	43908	43910	43910	43911	43911	43913	43913
43914	43914	43916	43916	43917	43917	43919	43919	43920	43920	43922	43922
43923	43923	43926	43926	43927	43927	43929	43929	43930	43930	43932	43932
43933	43933	43935	43935	43936	43936	43938	43938	43939	43939	43941	43941
43942	43942	43944	43944	43945	43945	43947	43947	43948	43948	43950	43950
43951	43951	43953	43953	43954	43954	43956	43956	43957	43957	43959	43959
43960	43960	43962	43962	43963	43963	43965	43965	43966	43966	43968	43968
43969	43969	43971	43971	43972	43972						

IE.CM.TRAP	38629 #	44029	44029	44030	44030	44032	44032	44033	44033	44036	44036	44037
	44037	44039	44039	44040	44040							
IE.CONSOLE.INT	37102 #											
IE.CONSOLE.INT2	37105	38200 #										
IE.CORRECTED.MEM	37117 #											
IE.CORRECTED.MEM2	37121	37583 #										
IE.CS.PARITY	37632 #											

```
IE.EXCEP01              37182   37192 #
IE.EXCEP02              37196   37202 #
IE.FAULT                20901   20930   37185 # 37878   38579   38772   39181   39191
IE.FLDBZ.FAULT          13048   14523   38743 #
IE.FLOV.FAULT           11820   12580   38737 # 47810
IE.FLT.DEC.DBZ          27728   29228   29370   38544 #
IE.FLT.FAULT            38740   38746   38752   38754 #
IE.FLUN.FAULT           12642   38749 #
IE.FPA.FAULT            38714 #
IE.FPA.FAULT.TRAP       38534   38726 #
IE.FPD.PACK             37868   38767   38928   38997 # 39158
IE.FPTCR                38531 # 38717
IE.GET.PC               37357   37365   37434 #
IE.ICR.DOSERV           38804 #
IE.ICR.DOSERV2          38807   38814 #
IE.ICR.INC              38827   38842 #
IE.ICR.SERV             8036    38817   38820 # 38898   38906
IE.ICR.SERV2            35321   35349   35978   38825 #
IE.ICR.SR               38830 # 38837   38845
IE.ICR.VP               38834   38848 # 38856
IE.INDEX.RANGE          21055   21064   38549 #
IE.INST.FAULT           38699   38706   38712   38730   38757   38759 #
IE.INT.DBZ              9329    9523    38537 #
IE.INT.TIMER            37112 #
IE.INT.TIMER2           37115   38062 #
IE.INT.WIDE.BRANCH      37095 # 38933   39174
IE.INTERRUPT            37215 # 37492   37537   37560   37586   38065   38174   38214   38229
IE.IRD1.ERROR           37644 #
IE.ISTACK               37275   37285 #
IE.KSNV                 37200   37231   38256 #
IE.KSNV.10              38264   38273 #
IE.KSTACK               37237   37263 #
IE.LOAD.PC              37372 # 38011   38487   38572   38664
IE.LOAD.PC01            36361   37377 #
IE.MACH.CHECK01         37873   37906 #
IE.MACH.CHECK02         37882   37956 #
IE.MACH.CHK.FAULT       37870 # 39176
IE.MACH.CHK.RBACK       37835   37850   37860 #
IE.MEM.ERR10            37629   37635   37641   37647   37652   37655 #
IE.MEM.ERR20            37848 # 37857
IE.MEM.ERROR            37626 #
IE.OPCOD.CUST           38709 #
IE.OPCOD.DEC            9849    9849    9849    9856    9856    9856    9856    9863    9863    9863    9863
                        9870    9870    9870    9877    9877    9877    9877    9884    9884    9884    9884
                        9891    9891    9891    9897    9897    9897    9897    9904    9904    9904    9904
                        9911    9911    9911    9918    9918    9918    9918    9925    9925    9925    9925
                        9932    9932    9932    9939    9939    9939    9939    9946    9946    9946    9946
                        9952    9952    9952    9959    9959    9959    9959    9966    9966    9966    9966
                        9973    9973    9973    9980    9980    9980    9980    9987    9987    9987    9987
                        9994    9994    9994    10001   10001   10001   10001   10007   10007   10007   10007
                        10014   10014   10014   10021   10021   10021   10021   10028   10028   10028   10028
                        10035   10035   10035   10042   10042   10042   10042   10049   10049   10049   10049
                        10056   10056   10056   10062   10062   10062   10062   10069   10069   10069   10069
                        10076   10076   10076   10083   10083   10083   10083   10090   10090   10090   10090
                        10097   10097   10097   10104   10104   10104   10104   10111   10111   10111   10111
```

Cross Reference Listing - Field Names and Defined Values

10117	10117	10117	10117	10124	10124	10124	10124	10131	10131	10131	10131
10138	10138	10138	10138	10145	10145	10145	10145	10152	10152	10152	10152
10159	10159	10159	10159	10166	10166	10166	10166	10172	10172	10172	10172
10179	10179	10179	10179	10186	10186	10186	10186	10193	10193	10193	10193
10200	10200	10200	10200	10207	10207	10207	10207	10214	10214	10214	10214
10221	10221	10221	10221	10227	10227	10227	10227	10234	10234	10234	10234
10241	10241	10241	10241	10248	10248	10248	10248	10255	10255	10255	10255
10262	10262	10262	10262	10269	10269	10269	10269	10276	10276	10276	10276
10282	10282	10282	10282	10289	10289	10289	10289	10296	10296	10296	10296
10303	10303	10303	10303	10310	10310	10310	10310	10317	10317	10317	10317
10324	10324	10324	10324	10331	10331	10331	10331	10337	10337	10337	10337
10344	10344	10344	10344	10351	10351	10351	10351	10358	10358	10358	10358
10365	10365	10365	10365	10372	10372	10372	10372	10379	10379	10379	10379
10386	10386	10386	10386	10392	10392	10392	10392	10399	10399	10399	10399
10406	10406	10406	10406	10413	10413	10413	10413	10420	10420	10420	10420
10427	10427	10427	10427	10434	10434	10434	10434	10441	10441	10441	10441
10447	10447	10447	10447	10454	10454	10454	10454	10461	10461	10461	10461
10468	10468	10468	10468	10475	10475	10475	10475	10482	10482	10482	10482
10489	10489	10489	10489	10496	10496	10496	10496	10502	10502	10502	10502
10509	10509	10509	10509	17645	17645	17645	17645	17652	17652	17652	17652
17659	17659	17659	17659	17666	17666	17666	17666	17673	17673	17673	17673
17680	17680	17680	17680	17691	17691	17691	17691	17697	17697	17697	17697
17711	17711	17711	17711	17718	17718	17718	17718	17725	17725	17725	17725
17732	17732	17732	17732	17739	17739	17739	17739	17746	17746	17746	17746
17752	17752	17752	17752	17759	17759	17759	17759	17766	17766	17766	17766
17773	17773	17773	17773	17780	17780	17780	17780	17787	17787	17787	17787
17794	17794	17794	17794	17801	17801	17801	17801	17807	17807	17807	17807
17814	17814	17814	17814	17821	17821	17821	17821	17828	17828	17828	17828
17835	17835	17835	17835	17842	17842	17842	17842	17849	17849	17849	17849
17856	17856	17856	17856	17862	17862	17862	17862	17869	17869	17869	17869
17880	17880	17880	17880	17887	17887	17887	17887	17894	17894	17894	17894
17901	17901	17901	17901	17908	17908	17908	17908	17914	17914	17914	17914
17939	17939	17939	17939	17946	17946	17946	17946	17953	17953	17953	17953
17960	17960	17960	17960	17966	17966	17966	17966	17973	17973	17973	17973
18777	18777	18777	18777	18784	18784	18784	18784	18791	18791	18791	18791
18798	18798	18798	18798	18805	18805	18805	18805	18812	18812	18812	18812
18819	18819	18819	18819	19744	19744	19744	19744	19751	19751	19751	19751
19758	19758	19758	19758	19765	19765	19765	19765	19772	19772	19772	19772
19779	19779	19779	19779	19786	19786	19786	19786	19792	19792	19792	19792
19799	19799	19799	19799	19806	19806	19806	19806	19813	19813	19813	19813
19820	19820	19820	19820	19827	19827	19827	19827	19834	19834	19834	19834
19841	19841	19841	19841	19847	19847	19847	19847	19856	19856	19856	19856
19865	19865	19865	19865	19872	19872	19872	19872	19879	19879	19879	19879
19886	19886	19886	19886	19893	19893	19893	19893	19899	19899	19899	19899
19906	19906	19906	19906	19915	19915	19915	19915	19924	19924	19924	19924
19931	19931	19931	19931	19938	19938	19938	19938	19945	19945	19945	19945
19951	19951	19951	19951	19958	19958	19958	19958	19965	19965	19965	19965
19972	19972	19972	19972	19979	19979	19979	19979	19986	19986	19986	19986
19993	19993	19993	19993	20000	20000	20000	20000	20006	20006	20006	20006
20013	20013	20013	20013	20825	20825	20825	20825	20834	20834	20834	20834
20843	20843	20843	20843	20959	22108	22108	22108	22108	22115	22115	22115
22115	22122	22122	22122	22122	22129	22129	22129	22129	22136	22136	22136
22136	22143	22143	22143	22143	22150	22150	22150	22150	22156	22156	22156
22156	22163	22163	22163	22163	22170	22170	22170	22170	22177	22177	22177
22177	22184	22184	22184	22184	22191	22191	22191	22191	22198	22198	22198

```
                    22198   22205   22205   22205   22205   33286   33286   33286   33286   34716   34856   36366
                    36366   36366   36366   36373   36373   36373   36373   36380   36380   36380   36380   36387
                    36387   36387   36387   36394   36394   36394   36394   36401   36401   36401   36401   36970
                    36974   36978   36983   36983   36983   36983   36993   36993   36993   36993   37003   37003
                    37003   37003   37013   37013   37013   37013   38703 #  39228   39280   39520   39520   39520
                    39520   39521   39521   39521   39521   39527   39527   39527   39527   39528   39528   39528
                    39528   39534   39534   39534   39534   39541   39541   39541   39541   39542   39542   39542
                    39542   39548   39548   39548   39548   39549   39549   39549   39549   39555   39555   39555
                    39555   39556   39556   39556   39556   39561   39561   39561   39561   39562   39562   39562
                    39562   39568   39568   39568   39568   39569   39569   39569   39569
IE.OPER.ABORT       38719 #
IE.OPER.FAULT       8648    10689   10760   10813   10864   10922   10988   11007   11098   11126   11230   11765
                    12066   12132   12151   12954   12974   13309   13352   13946   13965   15348   15414   16001
                    17000   17249   18375   18476   18479   18491   18494   18506   18509   18682   18739   18743
                    19160   20180   20576   21408   21608   21669   21682   21702   21791   21871   21893   21955
                    21962   21989   22017   25617   26120   26469   28494   29611   29825   30233   30237   30464
                    30502   30513   30704   30717   30738   30750   30775   30790   31116   31464   33471   34600
                    35666   36180   36726   36752   38725 #  39351   39360   39390   39394   47466   47626   47676
IE.PACK.BRANCH      39170 #
IE.PACK.DONE        16125   16129   17156   17165   17589   22103   24689   24698   24722   24963   24980   25178
                    25183   25258   25263   28946   28983   29219   29361   29403   30012   30050   31182   31865
                    31933   34624   39029   39143 #
IE.PACK.FAULT       39163   39187 #
IE.PACK.OR.RBACK    38390   38405   38434   38444   38473   39016   39049   39150 #
IE.PC.CHECK         39479   39482 #
IE.POWER.FAIL       37133 #
IE.POWER.FAIL2      37137   37534 #
IE.PUSH.PSL.PC      37283   37328 #
IE.RAISE.IPL        37375   37382 #
IE.RBS.LOOP         38953   38959 #  38971   38975
IE.RBS.RBACK        22099   37864   38763   38924   38950 #  39154
IE.RBS.UNDO         38962   38968 #
IE.READ.ACV         38364 #  39955
IE.READ.TNV         38450 #  39944
IE.REI              39328 #  39535   39535   39535   39535
IE.REI.10           39357 #  39776
IE.REI.20           39355   39362 #
IE.REI.50           39365   39379 #  39398
IE.REI.60           39382   39401 #
IE.REI.70           39404   39408   39412   39416   39420   39425 #
IE.REI.80           39427   39432 #
IE.REI.90           39442   39446   39450   39454   39472 #
IE.RESTORE.IPL      37394 #
IE.RET.TO.CN.BT.MV  37669 #  38409   38477
IE.SAVE.SP          37267   37289   37292 #
IE.SERV.FIX.VA      38894   38908 #
IE.SERV.IP          38913   38918   38920 #
IE.SERV.IP.TS       35540   36039   38881 #
IE.SERV.IP.TS2      14088   14118   14466   14484   15742   16563   17045   17290   17504   22440   23026   23163
                    23315   23458   23739   23853   24052   24140   24210   24347   24468   25665   25811   26115
                    26456   26598   27091   27320   27459   27469   27710   27765   28044   28178   28212   28298
                    28432   28466   28755   29504   29632   30685   30869   30952   31138   31448   31474   31608
                    31639   32416   32541   32742   32819   33567   34398   34455   35576   35603   36320   36692
                    38884   38891 #  40061   40146   40267   40408   47726
IE.SOFT.INT         37096 #
```

```
IE.SOFT.INT3            37144   37477 #
IE.SOFT.IPL             37099   37488   37507 # 39386
IE.TNV                  21225   36509   38467   38470 # 40696   40759   40818   40878
IE.TNV.CHK.CONSOLE      38461   38464 # 39890   39980
IE.TNV.RBACK            20150   38441 # 40417   40436
IE.TP.FAULT             38575 #
IE.TRAP                 37179 # 38563
IE.TU58                 38154 #
IE.TU58.CONT            38032 # 38157
IE.UNIBUS.INT           37107 #
IE.UNIBUS.INT2          37110   38144 #
IE.UNIBUS.INT5          38092   38113   38169   38171 #
IE.UNUSED.CS            36101   36106   37649 # 48084   48087   48090   48093   48096   48099
IE.WCS.BASE             37244 # 38303
IE.WRITE.MSCR           37962   37965   37969   37973   37977   38031 #
IE.WRITE.MSCR.0         37986   38037 #
IE.WRT.BUS.ERR          37127 # 37432
IE.WRT.BUS.ERR2         37131   37557 #
IE.WRT.BUS.EXCEP        37423   37429 #
IL.ADAWIMEM             8638 #  9853    9853    9853    9853    9853    9853
IL.ADAWIMEM2            8645    8650 #
IL.ADAWIREG             8633 #  9852    9852    9852    9852    9852
IL.ADD2.B.W.L.MEM       8592 #  9860    9860    9860    9860    9860    9874    9874    9874    9874    9874
                        9874    9888    9888    9888    9888    9888    9888
IL.ADD2.B.W.L.REG       8582 #  9859    9859    9859    9859    9873    9873    9873    9873    9887    9887    9887
                        9887
IL.ADD3.B.W.L.MEM       8599 #  9867    9867    9881    9881    9895    9895
IL.ADD3.B.W.L.REG       8593 #  9867    9867    9867    9867    9881    9881    9881    9881    9895    9895    9895
                        9895
IL.ADWCMEM              8555 #  9901    9901    9901    9901    9901    9901
IL.ADWCREG              8550 #  9900    9900    9900    9900    9900
IL.ASH                  9664 #  9908    9908    9908    9908    9908    9908    9915    9915    9915    9915    9915
                        9915
IL.ASHL                 9674 #
IL.ASHL.A               9677    9684 #  9690
IL.ASHL.B               9694    9700 #
IL.ASHL.C               9707    9714 #
IL.ASHLWRTQ             9698    9703    9711    9717    9723 #
IL.ASHQ                 9679 #
IL.ASHQ.A               9682    9735 #
IL.ASHQ.A1              9737    9758 #
IL.ASHQ.B               9739 #
IL.ASHQ.B1              9743    9767 #
IL.ASHQ.C               9745 #
IL.ASHQ.C1              9750    9784 #
IL.ASHQ.C2              9738    9793 #  9813
IL.ASHQ.D               9752 #
IL.ASHQ.D1              9756    9802 #
IL.ASHQ.D2              9809    9823 #
IL.ASHQ.WRT.MSW         9796    9820    9817    9821    9826    9832 #  9844
IL.ASHQ.WRT.MSW.SETV    9830    9837 #
IL.BIC2.B.W.L.MEM       8853 #  9922    9922    9922    9922    9922    9936    9936    9936    9936    9936
                        9936    9950    9950    9950    9950    9950
IL.BIC2.B.W.L.REG       8848 #  9921    9921    9921    9935    9935    9935    9935    9949    9949    9949
                        9949
```

Cross Reference Listing - Field Names and Defined Values

```
IL.BIC3.B.W.L.MEM      8841 #  9929  9929  9943  9943  9956  9956
IL.BIC3.B.W.L.REG      8854 #  9929  9929  9929  9929  9943  9943  9943  9943  9956  9956  9956
                       9956
IL.BIS2.B.W.L.MEM      8809 #  9963  9963  9963  9963  9963  9963  9977  9977  9977  9977  9977
                       9977    9991  9991  9991  9991  9991  9991
IL.BIS2.B.W.L.REG      8804 #  9962  9962  9962  9962  9976  9976  9976  9976  9990  9990  9990
                       9990
IL.BIS3.B.W.L.MEM      8817 #  9970  9970  9984  9984  9998  9998
IL.BIS3.B.W.L.REG      8810 #  9970  9970  9970  9970  9984  9984  9984  9984  9998  9998  9998
                       9998
IL.BIT.B.W.L.MEM       8237 #  10005 10005 10005 10005 10005 10005 10011 10011 10011 10011 10011
                       10011   10018 10018 10018 10018 10018 10018
IL.BIT.B.W.L.REG       8242 #  10004 10004 10004 10004 10010 10010 10010 10010 10017 10017 10017
                       10017
IL.CLR.B.W.L           8519 #  10024 10024 10024 10024 10024 10024 10031 10031 10031 10031 10031
                       10031   10045 10045 10045 10045 10045 10045
IL.CLRQ                8524 #  10038 10038 10038 10038 10038 10038
IL.CMP.B.W.L.MEM       8258 #  10053 10053 10053 10053 10053 10053 10060 10060 10060 10060 10060
                       10060   10066 10066 10066 10066 10066 10066
IL.CMP.B.W.L.REG       8263 #  10052 10052 10052 10052 10059 10059 10059 10059 10065 10065 10065
                       10065
IL.CVT.BW.BL.WL        8330 #  10072 10072 10072 10072 10072 10072 10079 10079 10079 10079 10079
                       10079   10107 10107 10107 10107 10107 10107
IL.CVT.LB.LW.WB.MEM    8289 #  10087 10087 10087 10087 10087 10087 10094 10094 10094 10094 10094
                       10094   10101 10101 10101 10101 10101 10101
IL.CVT.LB.LW.WB.REG    8304 #  10086 10086 10086 10086 10093 10093 10093 10093 10100 10100 10100
                       10100
IL.CVTWB.STRIP         8297    8312  8318 #
IL.DEC.B.W.L           8706 #  10114 10114 10114 10114 10114 10114 10120 10120 10120 10120 10120
                       10120   10127 10127 10127 10127 10127 10127
IL.DIV2.B.W.L          9298 #  10135 10135 10135 10135 10135 10135 10149 10149 10149 10149 10149
                       10149   10163 10163 10163 10163 10163 10163
IL.DIV3.B.W.L          9303 #  10142 10142 10142 10142 10142 10142 10156 10156 10156 10156 10156
                       10156   10170 10170 10170 10170 10170 10170
IL.DIVB2               9267 #  10134 10134 10134 10134 10134 10134
IL.DIVB3               9283 #  10141 10141 10141 10141 10141 10141
IL.DIVFORK             9301    9320  9321 #
IL.DIVL2               9277 #  10148 10148 10148 10148 10148 10148
IL.DIVL3               9293 #  10155 10155 10155 10155 10155 10155
IL.DIVQUO              9315    9333  #10073 10073 10073 10073 10073 10073 10080 10080 10080 10080
                       10080   10130 10108 10108 10108 10108 10108 10108
IL.DIVSUB              9324    9418  9425  9549 #  42109
IL.DIVSUB.A            9554    9587  #9591 9608
IL.DIVSUB.A1           9596    9610  #
IL.DIVSUB.B            9565    9581  9621 #  9625  9642
IL.DIVSUB.B1           9630    9644  #
IL.DIVSUB.C            9570    9587  #9585
IL.DIVW2               9272 #  10162 10162 10162 10162 10162 10162
IL.DIVW3               9288 #  10169 10169 10169 10169 10169 10169
IL.EDIV                9374 #  10176 10176 10176 10176 10176 10176
IL.EDIVA               9383    9393  #
IL.EDIVB               9388    9413  #
IL.EDIVC               9405    9430  9435  9441  9443 #
IL.EDIVC1              9445    9457  #
IL.FDIVD               9420    9430  9449  9459  9474 #
```

```
IL.EDIVE          9479 #  9490
IL.EDIVF          9454    9465    9471    9477    9483 #
IL.EDIVG          9486    9492 #
IL.EDIVH          9502    9508 #
IL.EMUL           9016 #  10183   10183   10183
IL.EMUL1          9031    9034    9036 #
IL.EMUL2          9045    9053 #
IL.EMUL3          9050    9061 #
IL.EMUL4          9059    9069 #
IL.INC.B.W.L      8617 #  10189   10189   10189   10189   10189   10189   10196   10196   10196   10196   10196
                  10196   10203   10203   10203   10203   10203   10203
IL.MCOM.B.W.L.MEM 8360 #  10211   10211   10211   10211   10211   10211   10218   10218   10218   10218   10218
                  10218   10225   10225   10225   10225   10225   10225
IL.MCOM.B.W.L.REG 8365 #  10210   10210   10210   10210   10217   10217   10217   10217   10224   10224   10224
                  10224
IL.MNEG.B.W.L.MEM 8371 #  10231   10231   10231   10231   10231   10231   10238   10238   10238   10238   10238
                  10238   10245   10245   10245   10245   10245   10245
IL.MNEG.B.W.L.REG 8375 #  10230   10230   10230   10230   10237   10237   10237   10237   10244   10244   10244
                  10244
IL.MOV.B.W.L.MEM  8347 #  10280   10280   10280   10280   10280   10280   10286   10286   10286   10286   10286
                  10286   10300   10300   10300   10300   10300   10300   10307   10307   10307   10307   10307
                  10307   10314   10314   10314   10314   10314   10314   10321   10321   10321   10321   10321
                  10321   36398   36398   36398   36398   36398   36398
IL.MOV.B.W.L.REG  8353 #  10279   10279   10279   10279   10285   10285   10285   10285   10299   10299   10299
                  10299
IL.MOVA.B.W.L.MEM 8348 #  10252   10252   10252   10252   10252   10252   10259   10259   10259   10259   10259
                  10259   10273   10273   10273   10273   10273   10273
IL.MOVA.B.W.L.REG 8354 #  10251   10251   10251   10251   10258   10258   10258   10258   10272   10272   10272
                  10272
IL.MOVA.Q.MEM     8433 #  10266   10266   10266   10266   10266   10266
IL.MOVA.Q.REG     8438 #  10265   10265   10265   10265
IL.MOVQ           8401 #  9791    10292   10292   10292   10292   10293   10293   10293   10293   10293   10293   10293
IL.MOVQMEM        8409 #
IL.MOVQMEM.A      8411    8418 #
IL.MOVQREG        8413 #
IL.MOVZ.BW.BL.WL  8385 #  10306   10306   10306   10306   10306   10306   10313   10313   10313   10313   10313
                  10313   10320   10320   10320   10320   10320   10320
IL.MUL2.B.W.L     8895 #  10328   10328   10328   10341   10341   10341   10355   10355   10355
IL.MUL3.B.W.L     8901 #  10335   10335   10335   10348   10348   10348   10362   10362   10362
IL.MULA           8899    8907 #
IL.MULB           8924 #  8937
IL.MULC           8920    8934 #
IL.MULD           8916    8944 #
IL.MULSETV        8932    8948 #  9341    9515    9727    9840
IL.MULSUB.+LOOP   9170    9175    9180    9185    9187 #  9191
IL.MULSUB.-LOOP   9209    9214    9219    9224    9226 #  9230
IL.MULSUB.CAND+   9149    9154    9166 #
IL.MULSUB.CAND-   9159    9164    9205 #
IL.MULSUB.MDR_0   8910    9039    9145 #  21032   41954
IL.MULWRITE-Q     8939 #
IL.PUSHA.B.W.L    8453 #  10368   10368   10368   10368   10368   10368   10375   10375   10375   10375   10375
                  10375   10389   10389   10389   10389   10389   10389
IL.PUSHA.Q        8465 #  10382   10382   10382   10382   10382   10382
IL.PUSHL          8452 #  10395   10395   10395   10395   10395   10395
IL.ROTIMEM        8490 #  10403   10403
```

```
IL.ROTLREG          8504 #  10403   10403   10403   10403
IL.SBWCMEM          8729 #  10410   10410   10410   10410   10410   10410
IL.SEWCREG          8724 #  10409   10409   10409   10409
IL.SOR+.END+        9550 #  9559    9574
IL.SOR+.END-        9556 #
IL.SOR-.END+        9561 #
IL.SOR-.END-        9567 #
IL.SUB2.B.W.L.MEM   8681 #  10417   10417   10417   10417   10417   10417   10431   10431   10431   10431   10431
                    10431   10445   10445   10445   10445   10445   10445
IL.SUB2.B.W.L.REG   8676 #  10416   10416   10416   10416   10430   10430   10430   10430   10444   10444   10444
                    10444
IL.SUB3.B.W.L.MEM   8688 #  10424   10424   10438   10438   10451   10451
IL.SUB3.B.W.L.REG   8682 #  10424   10424   10424   10424   10438   10438   10438   10438   10451   10451   10451
                    10451
IL.TST.B.W.L        3276 #  10457   10457   10457   10457   10457   10457   10464   10464   10464   10464   10464
                    10464   10471   10471   10471   10471   10471   10471
IL.XOR2.B.W.L.MEM   8765 #  10479   10479   10479   10479   10479   10479   10493   10493   10493   10493   10493
                    10493   10506   10506   10506   10506   10506   10506
IL.XOR2.B.W.L.REG   8760 #  10478   10478   10478   10478   10492   10492   10492   10492   10505   10505   10505
                    10505
IL.XOR3.B.W.L.MEM   8773 #  10486   10486   10500   10500   10513   10513
IL.XOR3.B.W.L.REG   8766 #  10486   10486   10486   10486   10500   10500   10500   10500   10513   10513   10513
                    10513
IN.CLR.CACHE        5995 #  6007
IN.CLR.CACHE.ROUT   5923    5988 #  47426
IN.DEC.D            5998    6003 #
IN.FLAG2.NOT.SET    5968    5971 #
IN.INIT             5354    5432    5873 #  6690    6960    [illegible]
IN.IORESET          5945 #  5948
IN.PC_0             5878 #  5899    5934
IN.PSL.LONLIT       5876    5890 #
IN.VA_0             5884 #  5992    35516
LS.LDPCTX           36665 # 36984   36984   36984   36984
LS.LDPCTX.GPRS.0    36694 # 36707
LS.LDPCTX.GPRS.1    36688   36697   36703 #
LS.LDPCTX.PODR      36701   36709 #
LS.LDPCTX.PSL       36773   36779 # 36803
LS.LDPCTX.PUSH      36777   36806 #
LS.LDPCTX.SUB1      36782   36809   36815 #
LS.MODE.CHECK       36675   36849   36962 #
LS.SVPCTX           36839 # 37014   37014   37014   37014
LS.SVPCTX.GPRS.2    36855 # 36867
LS.SVPCTX.GPRS.4    36858   36863 #
LS.SVPCTX.IPL       36896   36902 #
LS.SVPCTX.PC        36861   36868 #
LS.SVPCTX.SPS       36900   36924 #
MM.BUT.XB.TBMISS    39994 #
MM.GET.ACV          40065 # 40087   40150   40169   40191
MM.GET.BUT.FLUSH    40388 # 40445
MM.GET.BUT.PTE      38424   39998   40363 #
MM.GET.BUT.PTE05    40373 # 40411
MM.GET.BUT.PTE10    40377   40387   40396 # 40403
MM.GET.BUT.PTE20    40422   40427   40429 #
MM.GET.PTE          8123    40056   40142   40400   40480 #
MM.GET.PTE.P0       40484   40525 #
```

Cross Reference Listing - Field Names and Defined Values

```
MM.GET.PTE.P1            0489   40535 #
MM.GET.PTE.PX            0532   40538   40545 #
MM.GET.PTE.PX20         40557   40564 #
MM.GET.PTE.PX25         40574 # 40607
MM.GET.PTE.PX25.1       40581   40607 #
MM.GET.PTE.PX32         40568   40586   40618 #
MM.GET.PTE.SYS          40494   40500 #
MM.GET.READ.PTE         39920   40035   40046   40052 # 40930
MM.GET.READ.PTE20       40076   40081   40084 #
MM.GET.TNV              40068 # 40091   40154   40173   40195
MM.GET.WRITE.PCS33      40226   40229 #
MM.GET.WRITE.PTE        39976   40138 # 40902
MM.GET.WRITE.PTE20      40159   40166 #
MM.GET.WRITE.PTE25      40176 # 40237
MM.GET.WRITE.PTE30      40164   40168 #
MM.GET.WRITE.PTE31      40204 # 40215
MM.GET.WRITE.PTE32      40201   40210 #
MM.GET.WRITE.PTE33      40180   40218   40239 #
MM.GET.WRITE.PTE40      40096   40185   40208   40247 #
MM.GET.WRITE.PTE55      40253 # 40262
MM.GET.WRITE.PTE60      40250   40258 #
MM.GET.XB.PTE            9885   40026 #
MM.GET.XB.PTE10         40031 #
MM.PB.PROBE             29205 # 39726   39740
MM.PB.WRITE             29734 #
MM.PB.WRITE.UNLOCK      29720 #
MM.PB.WRITE10           29202   39629   39668   39753 #
MM.PB.WRITE15           39643   39657 #
MM.PB.WRITE5            29199 # 39721   39745
MM.PRB.READ.TBM         20957   21222   21240   36475   36492   40927 #
MM.PRB.WRITE.LONG       21439   21443   21450   21518   21532   40749 #
MM.PRB.WRITE.LONG.10    40767   40772 #
MM.PRB.WRITE.NR         21112   32791   40715   40775   40808 #
MM.PRB.WRITE.SIZ        21594   21675   21690   21798   21885   29620   40690 #
MM.PRB.WRITE.SIZ.00      9495   14758   16923   40685 #
MM.PRB.WRITE.SIZ.10     40706   40711 #
MM.PRB.WRITE1           36300   36304   40850 #
MM.PRB.WRITE1.05        40860 # 40882
MM.PRB.WRITE1.10        40858   40867 #
MM.PRB.WR1.TBM          20161   21123   36559   36576   39802   40693   40756   40815   40874   40899 #
MM.READ.ACV             39932   39952 # 39960
MM.READ.NO.ACCESS       38358   39927 #
MM.READ.TBMISS          39906 #
MM.READ.TBMISS.10       39909   39916 #
MM.READ.TNV             39925   39941 # 39949
MM.SAVE.INC.VA          39674   39684   39690 #
MM.SPTE.M.EQ.0          40213   40222   40295 #
MM.UA.READ              39642 #
MM.UA.WRITE             39681 #
MM.UA.WRITE.UNLOCK      39671 #
MM.WRITE.TBMISS         38346   39962 #
MM.WRITE.TBMISS.10      38457   39965   39972 #
MM.XB.ACV               39834   39893 #
MM.XB.ACV.-1            39810   39831 #
MM.XB.TBMISS            38385   39872 #
```

```
MM.XB.TBMISS.10          39875   39882 #
MM.XB.TNV                39806   39887 #
MM_PCALL_ACV_ENTRY       20783   20792   40699 #
MP.INVALIDATE.TB         35585   35599   35694 # 35698   35703   36679
MP.MFPR                   6832   35744 # 36397   36397   36397
MP.MFPR.ACCS             35865   36109 #
MP.MFPR.ASTLVL           35783   35911 #
MP.MFPR.BAD.IPR          35764   35787   35795   35806   35830   35846   35869   35877   35881   35889   35901   36177 #
MP.MFPR.BRANCH           35760   35768 #
MP.MFPR.BUILD.CMIERR     35919   35923   35927   35931   35935   35939   35946 #
MP.MFPR.CHK.CON          35751   35756   35771   35775   35791   35909   35941 # 35968   35987   35992   3c  ,   36077
                         36112   36116   36132   36143   36161
MP.MFPR.CMIERR           35798   35916 #
MP.MFPR.CONSOLE          35834   35838   35842   36085 #
MP.MFPR.CONSOLE.10       36083   36090 #
MP.MFPR.DONE             35944   36164 #
MP.MFPR.FPA              35739 # 36397   36397   3c397
MP.MFPR.ICCS             35802   35959 #
MP.MFPR.ICR              35810   35974 #
MP.MFPR.ICR10            35972   35980 #
MP.MFPR.IPL              35779   35903 #
MP.MFPR.MEMSCR           35850   35853   35857   35861   36094 #
MP.MFPR.MME              35873   36118 #
MP.MFPR.PMR              35893   36134 #
MP.MFPR.SID              35897   36144 #
MP.MFPR.TBDATA           35885   36123 #
MP.MFPR.TODR             35814   35994 #
MP.MFPR.TODR.20          36027   36035   36041 #
MP.MFPR.TODR.99          36021   36023 #
MP.MFPR.TODR.CHK         36031   36049 #
MP.MFPR.TODR.LP          36002 # 36047
MP.MFPR.TODR.RETRY       35989 # 36053
MP.MFPR.TU58             35818   35822   35826   36079 #
MP.MFPR.WRITE            35753 #
MP.MFPR.XTR.WRITE        35906 # 35914   36092   36097   36121
MP.MTPR                   6892   34950 # 36405   36405   36405   36405   36405   36405
MP.MTPR.ACCS             35115   35549 #
MP.MTPR.ASTLVL           35031   35249 #
MP.MTPR.BAD.IPR          34968   35043   35047   35059   35071   35087   35140   35148   35164   35173   35191   35196
                         35209   35218   35235   35244   35261   35304   35465   35474   35663 #
MP.MTPR.BRANCH           34984   34987 #
MP.MTPR.CADR             35103   35509 #
MP.MTPR.CADR05           35518 # 35532   35536
MP.MTPR.CADR10           35522   35529 #
MP.MTPR.CHK.CON          34961   34978   35027   35144   35180   35222   35248   35257   35288   35309   35338   35369
                         35376   35441   35446   35451   35485   35493   35547   35552   35554 # 35593   35608   35617
                         35625
MP.MTPR.CONSOLE          35083   35091   35443 # 35460
MP.MTPR.CSTS             35075   35379 # 35394
MP.MTPR.DONE             35160   35200   35231   35280   35423   35558   35653 #
MP.MTPR.EXIT             35636   35640   35642 # 37380   37397   37414
MP.MTPR.FLUSH.XB         35580   35614 #
MP.MTPR.GRP1             34957   34972 #
MP.MTPR.ICCS             35312 #
MP.MTPR.ICCS05           35051   35317 #
```

Cross Reference Listing - Field Names and Defined Values

```
MP.MTPR.ICCS'0          35315    35323 #
MP.MTPR.IORESET         35119    35560 # 35571
MP.MTPR.IORESET-1       35555 # 35567
MP.MTPR.MEMERR          35107    35111    35541 #
MP.MTPR.MEMSCR          35448 # 35503    35512    35526
MP.MTPR.MME             35123    35577 #
MP.MTPR.NICR            35340 #
MP.MTPR.NICR05          35055    35345 #
MP.MTPR.NICR10          35343    35351 #
MP.MTPR.NICR20          35366 #
MP.MTPR.P0BR            34992    35157 #
MP.MTPR.P1BR            35001    35183 #
MP.MTPR.PCBB            35019    35228 #
MP.MTPR.PXBR            35188 #
MP.MTPR.PXLR            34996    35005    35166 #
MP.MTPR.RESET.STEPC     6709     35647 #
MP.MTPR.RESV.OPER       35471 # 35498
MP.MTPR.SBR             35010    35202 #
MP.MTPR.SBR10           35193 # 35205
MP.MTPR.SCBB            35023    35237 #
MP.MTPR.SIRR            35035    35263 #
MP.MTPR.SIRR10          35276    35282 #
MP.MTPR.SISR            35291 # 35300
MP.MTPR.SISR05          35039    35296 #
MP.MTPR.SISR10          35294    35306 #
MP.MTPR.SLR             35014    35211 #
MP.MTPR.SPLIT           34964    34974    34980 #
MP.MTPR.TBCK            35152    35627 #
MP.MTPR.TBDATA          35136    35619 #
MP.MTPR.TBDR            35099    35495 #
MP.MTPR.TBDR.05         35500 # 35507
MP.MTPR.TBIA            35127    35582 #
MP.MTPR.TBIA20          5913     5920     35590    35596 #
MP.MTPR.TBIS            35132    35610 #
MP.MTPR.TODR            35064    35396 #
MP.MTPR.TODR.ADR        35412    35418    35425 #
MP.MTPR.TODR.LP         35415 # 35437
MP.MTPR.TU58            35067    35079    35382    35438 #
MP.MTPR.TXDB            35095    35453 #
MP.MTPR.TXDB.10         35456    35462 #
MP.MTPR.TXDB.20         35478    35490 #
MP.SELECT.IPR           34835 # 34859    34953    35747
MP.SELECT.IPR10         34838    34861 #
MP.SELECT.IPR20         34870 # 34887    34891
MP.SELECT.IPR50         34864    34894 #
MP.SELECT.IPR60         34904    34910 #
MP.SELECT.PI            34842    34846    34850    34853 #
MS.BICPSW               21392 # 22111    22111    22111    22111    22111    22111
MS.BISPSW               21382 # 22118    22118    22118    22118    22118    22118
MS.BIXPSW.MBZ           21386    21396    21405 #
MS.BPT                  20927 # 22123    22123    22123    22123
MS.COUNT.ONES           21128    21133    21245    21250    21308 # 21320
MS.COUNT.ONES10         21315    21322 #
MS.DEC.SP               21156 # 37994    37998    38002    38006    38035    38483
MS.HALT                 20948 # 22130    22130    22130    22130
```

Cross Reference Listing - Field Names and Defined Values

```
MS.HALT.MICRO          6128    6702    20970 # 36289   36337   37125   37256   37428   38272   47406
MS.INDEX               21003 # 22140   22140   22140   22140   22140   22140
MS.INDEX.50            21037   21043 #
MS.INDEX.60            21051   21058 #
MS.MOVPSL              21348 # 22166   22166   22166   22166   22166   22166
MS.NOP                 20912 # 22171   22171   22171   22171
MS.POPR                21199 # 22180   22180   22180   22180   22180   22180
MS.POPR.INNER          21269   21279 #
MS.POPR.LOOP           21265 # 21276   21283
MS.POPR.PRB            21210   21237 #
MS.POPR.PRB.OK         21234   21255   21257 #
MS.PUSHR               21097 # 22187   22187   22187   22187   22187   22187
MS.PUSHR.LOOP          21144   21150 #
MS.PUSHR.MASK          21154   21162 #
MS.PUSHR.PRB           21108   21120 #
MS.PUSHR.PRB.OK        21117   21140 #
MS.XFC                 20898 # 22211   22211   22211   22211   22212   22212   22212   22212
MV.BUS.TEST            46199 #
MV.CACHE.CHECK.FILL    47366   47373 #
MV.CACHE.CHECK.ZERO    47346   47350 #
MV.CACHE.PARITY        47142 #
MV.CACHE.TEST          47321 #
MV.CHECK.XB            46815   46832   46867 # 46986
MV.CLEANUP             47402   47417 #
MV.CLEAR.CACHE         47329 # 47338
MV.CLEAR.CACHE.LOOP    47334 #
MV.CLEAR.MME           47197   47211   47222   47237   47251   47262   47269 #
MV.CS.PARITY           47268   47291 #
MV.DSIZE.FIRST         47077   47085 #
MV.DSIZE.SETUP         47064 # 47070
MV DSIZE.TEST          47047 #
MV.END                 47349   47409 #
MV.ERR.605             46241 #
MV.ERR.606             46249 #
MV.ERR.60D             46303 #
MV.ERR.60E             46311 #
MV.ERROR               46243   46251   46305   46313   46391   46407   46416   46432   46441   46457   46475   46492
                       46569   46576   46583   46590   46595   46601   46608   46615   46622   46629   46635   46642
                       46648   46655   46662   46668   46675   46682   46689   46696   46744   46886   46893   46911
                       46922   46993   47003   47012   47022   47094   47101   47112   47154   47165   47176   47274
                       47295   47302   47356   47378   47391 #
MV.FILL.CACHE          47343 # 47371
MV.FILL.XB             46879   46942 #
MV.FINISH.SETUP        46171   46185 #
MV.GET.CLEAR.MEMSCR    47158   47169   47201   47215   47241   47255   47276 #
MV.LOAD.DSIZE          47080 # 47087   47105
MV.MBUS.FLOAT1         46284 # 46316
MV.MBUS.FLOAT1.05      46287   46293 #
MV.MBUS.FLOAT1.10      46280   46297 #
MV.MBUS.TEST           46231   46274 #
MV.PARITY.TEST         47133 #
MV.READ.CACHE          47363 # 47386
MV.RESTORE.GPR.10      46765 # 46781
MV.RESTORE.GPR.LP      46772   46778 #
MV.RESTORE.GPRS        46740   46761 #
```

Cross Reference Listing - Field Names and Defined Values

```
MV.RWBUS.FLOAT1          46224 # 46254
MV.RWBUS.FLOAT1.10       46219   46236 #
MV.RWBUS.FLOAT1.5        46227   46233 #
MV.SAVE.GPRS             46168 # 46183
MV.SAVE.GPRS.01          46166   46173 #
MV.SET.CRAR              8174    47428 #
MV.SP.ADR.LP             46523   46529   46534   46539   46541 # 46552   46557
MV.SP.CHECK.GPRS         46421 # 46437
MV.SP.CHECK.IPRS         46446 # 46462
MV.SP.CHECK.MTEMPS       46480 # 46496
MV.SP.CHECK.RTEMPS       46396 # 46412
MV.SP.DUAL.PORT          46714 #
MV.SP.DUAL.PORT.CHK      46736   46742 #
MV.SP.DUAL.PORT.LP       46721   46727 #
MV.SP.DUAL.PORT.READ     46725   46733 # 46749
MV.SP.DUAL.PORT.WRT      46718 # 46731
MV.SP.EXPLICIT.ADR       46468   46519 #
MV.SP.GPR.ADR            46531 #
MV.SP.GPR.ADR.ERR        46378   46414 #
MV.SP.GPR.ERROR          46424   46430 #
MV.SP.IPR.ADR            46536 #
MV.SP.IPR.ADR.ERR        46383   46439 #
MV.SP.IPR.ERROR          46449   46455 #
MV.SP.MTEMP.ERROR        46484   46490 #
MV.SP.MTEMP.LOOP         46464 # 46488
MV.SP.RTEMP.ADR          46526 # 46544
MV.SP.RTEMP.ADR.ERR      46373   46389 #
MV.SP.RTEMP.ERROR        46399   46405 #
MV.SP.TEST               46291   46342 #
MV.SP.TEST.GPRS          46375 #
MV.SP.TEST.IPRS          46380 #
MV.SP.TEST.MTEMPS        46387   46470 #
MV.SP.TEST.RTEMPS        46353   46370 # 46403   46428   46453
MV.SP.ZERO               46346 # 46368
MV.SP.ZERO.GPR           46349   46355 #
MV.STALL.FOR.CACHE       46176   46871   47053   47332   47361   47387 #
MV.TB.GRP0.PARITY        47179 #
MV.TB.GRP1.PARITY        47225 #
MV.TEST                  5349    5427    6698    7135    46143 #
MV.XB.PC.TEST            46835   46975 #
MV.XB.TEST               46776   46805 #
MV.XB.TEST.HIGH.BITS     46819   46825 #
MV.XB.TEST.LOOP1         46812 # 46823
MV.XB.TEST.LOOP2         46829 # 46839
MV.XB.WAIT               46949 # 46952
OLD161C                  29209   39799 #
OS.ADD                   10249   10249   10249   10249   10256   10256   10256   10256   10263   10263   10263   10263
                         10270   10270   10270   10270   10366   10366   10366   10366   10373   10373   10373   10373
                         10380   10380   10380   10380   10387   10387   10387   10387   15988   17072   17457   17705
                         17705   17705   17705   19987   19987   19987   19987   19994   19994   19994   19994   20826
                         20826   20826   20826   20828   20828   20828   20828   20628   20828   20837   20837   20837
                         20837   20837   20837   22144   22144   22174   22144   22146   22146   22146   22146   22146
                         22146   22151   22151   22151   22151   22[illegible]   22153   22153   22153   22153   22153   22157
                         22157   22157   22157   22159   22159   22159   22159   22159   22159   22192   22192   22192
                         22192   22199   22199   22199   22199   22206   22206   22206   22206   22419   22425   22498
```

Cross Reference Listing - Field Names and Defined Values

```
                        22527   22557   22576   22597   22605   22663   22699   22712   25600   25607   26090   26101
                        26418   26424   26440   29459   29810   30085   30091   30101   30176   30182   30197   30326
                        30338   30429   30440   31422   31442   33475   33484   33489   36627   44927 #
OS.ADD-AUTO.DEC         44946   45008 #
OS.ADD-AUTO.INC         44934   45029 # 45174   45281
OS.ADD-DISPLACEMENT     44956   44960   45019 #
OS.ADD-IMMEDIATE        44938   44994 #
OS.ADD-INC.PC           44998   45004 #
OS.ADD-INDEX            44989 # 45017
OS.ADD-READ.INC.4       44979   45035 #
OS.ADD-READ.SAV.EXIT    44970   44974   45023 #
OS.ADD-RELATIVE         44951   45014 #
OS.BOA                  44410   44467 # 44878   44986   45115   45416   45875
OS.BOA-EXIT             44485   44493   44497   44501   44520 # 44529   44539   44545   44759   44767   44771   44775
OS.BOA-EXIT.QUAD        44751   44796 #
OS.BOA-INC              44473   44525 # 44747
OS.BOA-INDEX-IMMED      44477   44548 # 44572   44662
OS.BOA-QUAD             44741 # 45503   45649   45759   45969   46066
OS.BOA-READ             44505   44508   44536 # 44779   44782
OS.BOA-READ.INC         44512   44541 # 44786
OS.BOA-RELATIVE         44489   44531 # 44763
OS.BOA-WRT1             44562 # 45235
OS.BOA-WRT1-EXIT        44580   44589   44616 # 44635   44641
OS.BOA-WRT1-INC         44568   44621 #
OS.BOA-WRT1-READ        44601   44604   44632 #
OS.BOA-WRT1-READ.INC    44608   44637 #
OS.BOA-WRT1-RELATIVE    44585   44627 #
OS.BOA-WRT2             44652 # 45332
OS.BOA-WRT2-EXIT        44670   44679   44683   44687   44706 # 44719   44724   44730   44799
OS.BOA-WRT2-INC         44658   44715 #
OS.BOA-WRT2-READ        44691   44694   44721 #
OS.BOA-WRT2-READ.INC    44698   44726 #
OS.BOA-WRT2-RELATIVE    44675   44711 #
OS.DMOD                 13160   13456   13768   14311   45700 #
OS.DMOD-AUTO.INC        45708   45769 # 46014
OS.DMOD-DEC.VA          45781   45783 #
OS.DMOD-INDIRECT        45742   45747   45787 #
OS.DMOD-READ.INC.4      45752   45797 # 46059
OS.DMOD-READ1           45718   45723   45728   45733   45737   45762 # 45795
OS.DMOD-READ2           45767   45775   45777 # 46071   46079
OS.DMOD-VA_MDR          45790   45792 # 45802   46085
OS.DRED                 11116   13236   13788   14331   14956   15478   17646   17646   17646   17646   17660   17660
                        17667   17667   17692   17692   17726   17726   17733   17733   17740   17740   17747   17747
                        17795   17795   17822   17822   17829   17829   17850   17850   17863   17863   17863   17863
                        17881   17881   17881   17881   17895   17895   17902   17902   17922   17922   17940   17940
                        17947   17947   17967   17967   17967   17967   45587 #
OS.DRED-IMMEDIATE       45599   45680 #
OS.DRED-READ            45632   45637   45661 #
OS.DRED-REG             45590   45655 # 45703
OS.DRED-UNPACK          45604   45667 #
OS.FIDMOD               17662   17824   17897   17942   46006 #
OS.FIDMOD-INDIRECT      46049   46054   46081 #
OS.FIDMOD-READ1         46025   46030   46035   46040   46044   46073 #
OS.FIDRED               16965   17660   17660   17667   17667   17669   17692   17692   17694   17726   17726   17733
                        17733   17740   17740   17747   17747   17795   17795   17822   17822   17829   17829   17831
```

```
                    17850   17850   17895   17895   17902   17902   17904   17922   17922   17940   17940   17947
                    17947   17949   45896 #
OS.FIDRED-REG       45900   45975 # 46009
OS.FIDRED.IMMED     45910   45980 #
OS.FIDRED.READ2     45983   45985 #
OS.FRED             12042   14652   17653   17653   17653   17653   17655   17655   17655   17655   17655   17655
                    17674   17674   17681   17681   17683   17683   17683   17698   17698   17700   17700   17700
                    17753   17753   17760   17760   17767   17767   17774   17774   17802   17802   17836   17836
                    17843   17843   17845   17845   17845   17857   17857   17870   17870   17870   17870   17888
                    17888   17888   17888   17909   17909   17915   17915   17917   17917   17917   17929   17929
                    17929   17929   17954   17954   17961   17961   17974   17974   17974   17974   45352 #
OS.FRED-UNPACK      45370   45422 #
OS.MOD              9271    9276    9281    9859    9859    9873    9873    9887    9887    9900    9900    9921
                    9921    9935    9935    9949    9949    9962    9962    9976    9976    9990    9990    10112
                    10112   10112   10112   10118   10118   10118   10118   10125   10125   10125   10125   10187
                    10187   10187   10187   10194   10194   10194   10194   10201   10201   10201   10201   10327
                    10327   10327   10327   10327   10327   10340   10340   10340   10340   10340   10340   10354
                    10354   10354   10354   10354   10354   10409   10409   10416   10416   10430   10430   10444
                    10444   10478   10478   10492   10492   10505   10505   12035   15301   17676   17676   17676
                    17676   17838   17838   17838   17838   17911   17911   17911   17911   17956   19280   19768
                    19768   19775   19775   44817 #
OS.MOD-AUTO.INC     44826   44886 # 45823
OS.MOD-READ.EXIT    44857   44881 # 44909   45854
OS.MOD-READ.INC.4   44871   44910 # 45868
OS.MOD-READ.SAV     44861   44866   44900 # 45858   45863
OS.MOD-READ.SAV.EXIT 44837  44842   44847   44852   44893 # 45834   45839   45844   45849
OS.MOD-VA_MDR       44904   44906 # 44915
OS.QRED             9914    9914    9914    9914    9914    9914    10175   10175   10175   10175   10175   10175
                    10290   10290   10290   10290   45440 #
OS.QRED-AUTO.INC    45448   45519 # 45595   45905
OS.QRED-IMMEDIATE   45452   45527 #
OS.QRED-INDIRECT    45554   45556 # 45565   45664
OS.QRED-LITERAL     45457   45531 #
OS.QRED-MDR_0       45534   45567 # 45670
OS.QRED-READ.INC.4  45496   45561 # 45642   45961
OS.QRED-READ.IND    45486   45491   45550 # 45950   45956
OS.QRED-READ.SAV.INC 45462  45467   45472   45477   45481   45536 # 45922   45928   45934   45939   45944
OS.QRED-READ1       45506 # 45559   45608   45613   45618   45623   45627
OS.QRED-READ2       45511   45525   45542   45544 # 45654   45974
OS.QRED-REG         45443   45513 #
OS.RED              9019    9095    9287    9292    9297    9850    9850    9850    9850    9857    9857    9857
                    9857    9864    9864    9864    9864    9866    9866    9866    9866    9866    9866    9871
                    9871    9871    9871    9878    9878    9878    9878    9880    9880    9880    9880    9880
                    9880    9885    9885    9885    9885    9892    9892    9892    9892    9894    9894    9894
                    9894    9894    9894    9898    9898    9898    9898    9905    9905    9905    9905    9907
                    9907    9907    9907    9907    9907    9912    9912    9912    9912    9919    9919    9919
                    9919    9926    9926    9926    9926    9928    9928    9928    9928    9928    9928    9933
                    9933    9933    9933    9940    9940    9940    9940    9942    9942    9942    9942    9942
                    9942    9947    9947    9947    9947    9953    9953    9953    9953    9955    9955    9955
                    9955    9955    9955    9960    9960    9960    9960    9967    9967    9967    9967    9969
                    9969    9969    9969    9969    9969    9974    9974    9974    9974    9981    9981    9981
                    9981    9983    9983    9983    9983    9983    9983    9988    9988    9988    9988    9995
                    9995    9995    9995    9997    9997    9997    9997    9997    9997    10002   10002   10002
                    10002   10004   10004   10008   10008   10008   10008   10010   10010   10015   10015   10015
                    10015   10017   10017   10050   10050   10050   10050   10052   10052   10057   10057   10057
```

Cross Reference Listing - Field Names and Defined Values

10057	10059	10059	10063	10063	10063	10063	10065	10065	10070	10070	10070
10070	10077	10077	10077	10077	10084	10084	10084	10084	10091	10091	10091
10091	10098	10098	10098	10098	10105	10105	10105	10105	10132	10132	10132
10132	10139	10139	10139	10139	10146	10146	10146	10146	10153	10153	10153
10153	10160	10160	10160	10160	10167	10167	10167	10167	10173	10173	10173
10173	10180	10180	10180	10180	10182	10182	10182	10182	10182	10182	10208
10208	10208	10208	10215	10215	10215	10215	10222	10222	10222	10222	10228
10228	10228	10228	10235	10235	10235	10235	10242	10242	10242	10242	10277
10277	10277	10277	10283	10283	10283	10283	10297	10297	10297	10297	10304
10304	10304	10304	10311	10311	10311	10311	10318	10318	10318	10318	10325
10325	10325	10325	10332	10332	10332	10332	10334	10334	10334	10334	10334
10334	10338	10338	10338	10338	10345	10345	10345	10345	10347	10347	10347
10347	10347	10347	10352	10352	10352	10352	10359	10359	10359	10359	10361
10361	10361	10361	10361	10361	10393	10393	10393	10393	10400	10400	10400
10400	10402	10402	10402	10402	10402	10402	10407	10407	10407	10407	10414
10414	10414	10414	10421	10421	10421	10421	10423	10423	10423	10423	10423
10423	10428	10428	10428	10428	10435	10435	10435	10435	10437	10437	10437
10437	10437	10437	10442	10442	10442	10442	10448	10448	10448	10448	10450
10450	10450	10450	10450	10450	10455	10455	10455	10455	10462	10462	10462
10462	10469	10469	10469	10469	10476	10476	10476	10476	10483	10483	10483
10483	10485	10485	10485	10485	10485	10485	10490	10490	10490	10490	10497
10497	10497	10497	10499	10499	10499	10499	10499	10499	10503	10503	10503
10503	10510	10510	10510	10510	10512	10512	10512	10512	10512	10512	14942
16430	16902	17313	17318	17335	17443	17451	17674	17674	17681	17681	17683
17698	17698	17700	17712	17712	17712	17712	17719	17719	17719	17719	17753
17753	17760	17760	17767	17767	17774	17774	17781	17781	17781	17781	17788
17788	17788	17788	17802	17802	17808	17808	17808	17808	17815	17815	17815
17815	17836	17836	17843	17843	17845	17852	17852	17852	17857	17857	17859
17859	17859	17859	17859	17859	17909	17909	17915	17915	17917	17931	17931
17931	17931	17931	17931	17954	17954	17961	17961	17963	18388	18448	18455
18513	18517	18524	18652	18778	18778	18778	18778	18780	18780	18780	18780
18780	18780	18785	18785	18785	18785	18787	18787	18787	18787	18787	18787
18792	18792	18792	18792	18794	18794	18794	18794	18794	18794	18799	18799
18799	18799	18801	18801	18801	18801	18801	18801	18806	18806	18806	18806
18808	18808	18808	18808	18808	18808	18813	18813	18813	18813	18815	18815
18815	18815	18815	18815	18820	18820	18820	18820	18822	18822	18822	18822
18822	18822	19515	19745	19745	19745	19745	19747	19747	19747	19747	19747
19747	19752	19752	19752	19752	19754	19754	19754	19754	19754	19754	19759
19759	19759	19759	19761	19761	19761	19761	19761	19761	19766	19766	19766
19766	19773	19773	19773	19773	19780	19780	19780	19780	19787	19787	19787
19787	19793	19793	19793	19793	19800	19800	19800	19800	19807	19807	19807
19807	19814	19814	19814	19814	19821	19821	19821	19821	19828	19828	19828
19828	19880	19880	19880	19880	19887	19887	19887	19887	19966	19966	19966
19966	19968	19968	19968	19968	19968	19968	19973	19973	19973	19973	19975
19975	19975	19975	19975	19975	19980	19980	19980	19980	19982	19982	19982
19982	19982	19982	20835	20835	20835	20835	21007	21012	21017	22109	22109
22109	22109	22116	22116	22116	22116	22137	22137	22137	22137	22139	22139
22139	22139	22139	22139	22178	22178	22178	22178	22185	22185	22185	22185
22509	22516	22569	22583	22611	22654	22705	25413	25413	25413	25413	25420
25420	25420	25420	25427	25427	25427	25427	25434	25434	25434	25434	25441
25441	25441	25441	25448	25448	25448	25448	25455	25455	25455	25455	25461
25461	25461	25461	25468	25468	25468	25468	25475	25475	25475	25475	25482
25482	25482	25482	26095	26436	26452	30096	30187	30333	30435	31412	31427
31432	33252	33252	33252	33252	33259	33259	33259	33259	33266	33266	33266
33266	33273	33273	33273	33273	33280	33280	33280	33280	33287	33287	33287

Cross Reference Listing - Field Names and Defined Values

```
                                33287   33289   33289   33289   33289   33289   33289   33294   33294   33294   33294   33300
                                33300   33300   33300   33307   33307   33307   33307   33314   33314   33314   33314   33321
                                33321   33321   33321   33328   33328   33328   33328   33335   33335   33335   33335   33342
                                33342   33342   33342   33348   33348   33348   33348   33355   33355   33355   33355   34773
                                34773   34773   34773   36367   36367   36367   36367   36374   36374   36374   36374   36381
                                36381   36381   36381   36388   36388   36388   36388   36395   36395   36395   36395   36402
                                36402   36402   36402   36404   36404   36404   36404   36404   36404   36994   36994   36994
                                36994   36996   36996   36996   36996   36996   36996   37004   37004   37004   37004   37006
                                37006   37006   37006   37006   37006   44346 #
OS.RED-AUTO.INC                 44354   44421 # 45361
OS.RED-IMMED                    44358   44428 # 45365   45530
OS.RED-READ.EXIT                44388   44413 # 44449   45394
OS.RED-READ.INC.4               44403   44451 # 45409
OS.RED-READ.SAV                 44393   44398   44440 # 45399   45404
OS.RED-READ.SAV.EXIT            44369   44374   44379   44384   44433 # 45375   45380   45385   45390
OS.RED-VA_MDR                   44444   44446 # 44456
OS.SOB                          20007   20007   20007   20007   20014   20014   20014   20014   45815 #
OS.SOB-READ.EXIT                45878 #
OS.VADD                         18368   18396   18672   18675   18679   19144   45054 #
OS.VADD-AUTO.INC                45063   45145 #
OS.VADD-DISPLACEMENT            45085   45089   45123 # 45143
OS.VADD-INC.PC                  45067   45140 #
OS.VADD-INDEX                   45080   45118 #
OS.VADD-READ.EXIT               45098   45103   45128 #
OS.VADD-READ.INC.4              45108   45133 #
OS.WRT1                         8333    8388    8498    8603    8692    8777    8821    8866    8905    8977    9100    9306
                                9378    9396    9401    9665    10022   10022   10022   10022   10029   10029   10029   10029
                                10036   10036   10036   10036   10043   10043   10043   10043   11253   11322   11558   11647
                                11903   13241   13793   14336   14662   14869   14875   14973   16735   16748   16804   16818
                                16860   16907   16970   18391   18436   18443   18498   18502   18521   21022   22164   22164
                                22164   22164   29921   36167   45002   45006   45166 # 45570   45683
OS.WRT1-ABSOLUTE                45203 #
OS.WRT1-AUTO.DEC                45183 #
OS.WRT1-DEFERRED                45240   45247   45249 # 45338
OS.WRT1-DISPLACEMENT            45197 #
OS.WRT1-READ                    45217   45222   45237 #
OS.WRT1-READ.INC.4              45227   45242 # 45325
OS.WRT1-RELATIVE                45190 #
OS.WRT2                         9024    9852    9852    10086   10086   10093   10093   10100   10100   10210   10210   10217
                                10217   10224   10224   10230   10230   10237   10237   10244   10244   10251   10251   10258
                                10258   10265   10265   10272   10272   10279   10279   10285   10285   10292   10292   10299
                                10299   17735   17735   17735   17755   17755   17755   17769   17769   17769   17776   17776
                                17776   17790   17790   17790   17804   17804   17804   17865   17865   17872   17872   17883
                                17883   17890   17890   22194   22194   22201   22201   22208   22208   45272 # 45292   45297
                                45302   45306
OS.WRT2-READ                    45315   45320   45334 #
PC.CALL-10                      20431 # 20438
PC.CALL-ACV                     20147   20171   20566   20776 #
PC.CALL-APPX.PROBE              20124   20135   20152 # 20816
PC.CALL-COUNT.ONES              20780   20788   20794 # 20806
PC.CALL-DEC.SP                  20240   20249   20258   20266   20275   20284   20293   20301   20310   20319   20328   20336
                                20345   20501 #
PC.CALL-DEC.SP1                 20230   20425   20455   20506 #
PC.CALL-DEC.SP2                 20434   20511 #
PC.CALL-DONE.REG                20221   20235   20412 #
```

Cross Reference Listing - Field Names and Defined Values

```
PC.CALL-EXIT                20484 # 20491   20495   20499
PC.CALL-FIND.REG            20213 # 20244   20253   20262   20270   20279   20288   20297   20305   20314   20323   20332
                            20340   21166
PC.CALL-FIX.PROBE           20801   20808 #
PC.CALL-MASK.MASK           20192   20205 #
PC.CALL-PUSH.PC             20415   20422 #
PC.CALL-PUSH.R0             20227 # 20350
PC.CALL-PUSH.R1             20237 # 20355
PC.CALL-PUSH.R10            20316 # 20399
PC.CALL-PUSH.R11            20325 # 20404
PC.CALL-PUSH.R12            20333 # 20409
PC.CALL-PUSH.R13            20342 #
PC.CALL-PUSH.R2             20246 # 20360
PC.CALL-PUSH.R3             20255 # 20365
PC.CALL-PUSH.R4             20263 # 20370
PC.CALL-PUSH.R5             20272 # 20375
PC.CALL-PUSH.R6             20281 # 20379
PC.CALL-PUSH.R7             20290 # 20384
PC.CALL-PUSH.R8             20298 # 20389
PC.CALL-PUSH.R9             20307 # 20394
PC.CALL-SAVE.SP.BITS        20450   20516 #
PC.CALL-TEST.IV.DV          20472   20477 #
PC.CALL-TNV                 20142 # 20165   20561
PC.CALLG                    20095 # 20829   20829   20829   20829   20829   20829
PC.CALLG-ALLIGN             20112   20121 # 20129
PC.CALLS                    20096 # 20838   20838   20838   20838   20838   20838
PC.CALLS-ALLIGN             20118   20132 # 20140
PC.REI                      20528 # 20844   20844   20844   20844
PC.RET-DONE.REG             20601 #
PC.RET-EXIT                 20754 # 20769
PC.RET-FIX.STACK            20604   20728 #
PC.RET-PROBE                20545   20550 # 20820
PC.RET-WHICH.INST           20731   20735   20739   20746   20748 #
PC.RET-WRITE.GPR            20614   20623   20633   20643   20653   20663   20672   20682   20692   20702   20712   20721
                            20771 #
PCS.IE.CHECK.FOR.CACHE      37701   37712   37727   37825 #
PCS.IE.TB.PE                37709   37716 #
PCS.IE.TB.PE.1              37723   37731 #
PCS.IE.TB.PE.2              37748   37761 #
PCS.IE.TB.PE.INVALIDATE     37752   37765   37770   37789   37796 #
PCS.IE.TB.PTE               37740   37785 #
PR.PROBER                   36463 # 36997   36997   36997   36997   36997   36997
PR.PROBEW                   36547 # 37007   37007   37007   37007   37007   37007
PR.PROBEX.ACV               36483   36500   36520 # 36567   36584
PR.PROBEX.FLUSH             36518   36523   36525 #
PR.PROBEX.OK                36504   36515 # 36588
PR.PROBEX.SUB               36466   36550   36612 #
PR.PROBEX.SUB.10            36636   36641 #
PR.TNV                      36479   36496   36506 # 36563   36580
QU.CHECK.ALIGN              21979   21986 # 28477
QU.CHECK.ALIGN.1            21991 # 22028
QU.INSQHI                   21582 # 22147   22147   22147   22147   22147   22147
QU.INSQTI                   21648 # 22154   22154   22154   22154   22154   22154
QU.INSQTI.10                21720   21728 #
QU.INSQUE                   21432 # 22160   22160   22160   22160   22160   22160
```

```
    QU.LOCK.CHECK           21973   28473 #
    QU.LOCKED.ALIGN         21976 # 28500
    QU.REMQHI               21760 # 22195   22195   22195   22195   22195   22195
    QU.REMQHI.10            21776   21784 #
    QU.REMQHI.20            21625   21817 #
    QU.REMQHI.REG           21756 # 22194   22194   22194   22194
    QU.REMQTI               21850 # 22202   22202   22202   22202   22202   22202
    QU.REMQTI.REG           21846 # 22201   22201   22201   22201
    QU.REMQUE.10            21513   21525 #
    QU.REMQUE.MEM           21515 # 22209   22209   22209   22209   22209   22209
    QU.REMQUE.REG           21510 # 22208   22208   22208   22208
    QU.RESV.EXIT            22096 # 26217
    QU.RESV.FIX             26208 # 31074
    QU.RESV.OPER            31065 # 39087
    QU.SET.LOCK             21585   21651   21764   21854   21949 #
    QU.UNLOCK               21734   21820   21824   21912   21920   22044 #
    QU.UNLOCK.10            21984   22025   22052 #
    QU.WRITE.TEMP1          21468   21472   21480 # 21543
    VF.CMPV                 18327 # 18781   18781   18781   18781   18781   18781
    VF.CMPZV                18312 # 16788   18788   18788   18788   18788   18788
    VF.CMPZV.1              18318 # 18335
    VF.EXTV                 18284 # 18795   18795   18795   18795   18795   18795
    VF.EXTZV                18271 # 18802   18802   18802   18802   18802   18802
    VF.FFC                  18214 # 18809   18809   18809   18809   18809   18809
    VF.FFC.SIZ0TO31         18231   18247 #
    VF.FFS                  18195 # 18816   18816   18816   18816   18816   18816
    VF.FFS.OR.FFCSIZ32      18208   18233 #
    VF.FFSFOUND.FFC0TO31    18238 # 18250
    VF.INSV                 18649 # 18823   18823   18823   18823   18823   18823
    VF.INSV.MEM             18660   18690 #
    VF.INSV.MEM.ALIND       18699   18705 #
    VF.INSV.MEM.UNALIND     18703   18712 #
    VF.INSV.MEM1            18708   18731 #
    VF.INSV.REG             18665   18736 #
    VF.INSV.REG1            18747   18759 #
    VF.INSV.REG2            18762   18767 #
    VF.INSV1                18656   18669 #
    VF.VSUB                 18199   18218   18275   18287   18316   18331   18361 #
    VF.VSUB.MEM             18364   18372   18393 #
    VF.VSUB.MEM2            18410   18421 #
    VF.VSUB.MEM3            18414   18433 #
    VF.VSUB.MEM4            18418   18445 #
    VF.VSUB.REG             18398 #
    VF.VSUB.REG1            18402   18457 #
    VF.VSUB.REG2            18461   18474 #
    VF.VSUB.REG3            18466   18489 #
    VF.VSUB.REG4            18471   18504 #
    VF.VSUB.REG5            18424   18431   18483   18487   18519 #
    VF.VSUB.SIZ0            18379   18386 #
PATCH                       3034 #  20146   20170   21665   21969   36147   37662   40200   40556   40656   44998
ROT                         3038 #  9168    9173    9178    9183    9189    9194    9198    9202    9207    9212    9217
                            9222    9228    9233    9237    9241    9453    9464    9470    9552    9563    9579    9589
                            9594    9599    9606    9612    9617    9623    9628    9633    9640    9646    9651    12849
                            13079   14207   14218   14233   14253   14460   14478   14670   15781   42118   42124   42133
    ASL.M.P                 3072 #  9693    9747    9808    17550   20183   20518   41321   41412
```

Cross Reference Listing - Field Names and Defined Values

ASL.R.7	3071 #	13342	13346	13374	13986							
ASL.R.P	3069 #	14512										
ASL.R.SIZ	3070 #	11458	11803	11816	14645	15161	16532	20767	20814	20819	27820	28240
	29902	33119	37479	44522	44533	44708	44713	44804	45233	45677	48027	
ASR.M.-P	3075 #	9686	9702	9710	9716	9741	9763	9795	9799	9825	9843	12410
	41327	41418	41433	41438								
ASR.M.3	3076 #	18395	18659	19047	19060	19070	47396					
ASR.M.P	3074 #	9773	9781	12389	12887	12891	13692	13706	18290	18334	35213	35961
BCDSWP	3079 #	27086	28713	28721	28733	28741	28749	29545	29549	32197	32214	32234
	32251	32270	32288	32535	32554	33182	33190	33198	33215	33218	33221	33229
	33232	33235	33530	36056	41746	41755	41766					
CLR1BM	3046 #	33014	35094									
CLR2BM	3047 #	32490	32941	32947	32978	37340						
CLR3BM	3048 #	28069										
CONX.SIZ	3100 #	5546	5552	6138	6166	6436	6498	6512	6516	6607	6656	6701
	6779	6787	6795	6814	6817	6843	6903	6910	6917	6922	7219	7317
	7325	7333	7341	7348	7356	7364	7386	7401	7413	7429	7441	7444
	7450	7462	7472	7485	7488	7495	7507	7510	7517	7529	7539	7795
	7814	7923	8194	8455	8467	16849	19639	19722	19738	20190	20195	20418
	20444	20475	20503	20508	20513	20530	20569	20734	20738	21147	21152	21158
	21272	21449	21531	21539	23085	23144	23190	23196	23201	23253	23295	23319
	23324	23329	23725	23821	23825	23829	23833	23838	24019	24027	24107	24115
	24644	27275	27298	27303	27737	27889	27964	27992	27995	28017	28078	29097
	30346	30359	30363	30967	31248	31548	31574	31673	33720	33726	33737	34036
	34185	34285	34297	34368	34446	34450	34509	36341	36357	36817	36878	36891
	37330	37344	37356	37364	37678	37980	37989	38539	38566	38614	38620	38658
	38733	38775	39166	39184	39335	39341	39507	40713	40718	40774	40778	42402
	42557	42579	42603	42652	42715	42772	42776	42782	42829	42845	42920	42925
	42931	42975	42991	43061	43066	43072	43116	43130	43166	44367	44424	44454
	44483	44527	44544	44550	44578	44623	44640	44668	44717	44729	44757	44798
	44835	44889	44913	45001	45010	45031	45038	45073	45136	45147	45185	45245
	45290	45373	45460	45523	45564	45607	45717	45772	45800	45832	45919	46023
	46394	46419	46444	47082								
CVTNP	3082 #	30229	30505	30708	30741	30768						
CVTPN	3080 #	30355	30923	30957	30962	30972	30977					
FPACK	3088 #	11824	11880	11888	12575							
FPLIT	3087 #	17321	17326	45424	45669							
GETEXP	3085 #	10683	10738	10745	10308	10909	10914	10976	10981	11092	11121	11224
	11550	11639	11760	11982	12051	12127	12838	12949	13056	13151	13227	13448
	13460	13761	13780	13861	14304	14323	14418	14655	14935	14959	15308	15313
	15393	15480	15724	15752	15786	16025	16039	16246	16419	16555	16654	
GETFPF	3086 #	11793	12061	12069	12135	12145	12154	12165	12957	12967	12977	12988
	13330	13336	13371	13981								
GETNIB	3078 #	6800	7113	7638	7686	7781	7786	7899	17527	35957	38952	38970
	38974											
MINUS1	3099 #	5467	5563	5947	5973	6889	7416	7584	8367	8619	8708	8897
	8903	8995	9034	9115	9269	9274	9279	9285	9290	9295	9603	9637
	9676	9681	9829	11319	11445	11561	11807	13715	14660	15799	15850	15916
	16211	16267	16510	16548	16567	16593	16994	17030	17052	17242	17303	17491
	18221	18685	18688	19388	19394	19448	19455	21142	21259	21267	22991	23056
	23110	23115	23119	23261	23265	23269	23526	23530	23535	23540	23544	23550
	23875	23880	23884	23888	23997	24004	24010	24088	24092	24189	24215	24219
	24238	24323	24327	24569	24575	24774	25075	25182	25601	25655	26088	26099
	26417	26423	26439	26641	27097	27136	27177	27182	27186	27258	27264	27271
	27356	27360	27747	27787	27791	27823	27959	28030	28347	29649	29654	30400

	30488	30521	31411	31421	31435	31522	32151	32719	33518	33948	33959	34161
	34354	34358	34553	34599	34696	36293	36295	36331	36469	36517	36553	36673
	38832	38844	39002	39006	39053	39060	39096	39103	39111	39117	40316	40521
	40705	40722	40766	40781	41024	41035	41042	41062	41073	41080	41457	41571
	41947	42081	42477	42492	46159	46466	46550	46555	46983	47184	47193	47233
OLITO	3110 #	5582	5698	6288	6302	6316	6330	6344	6358	6372	6400	6414
	15811	16622	20465	42044								
OLITO.PL43_WB	3093 #	18405	18692	19149								
OLITO.PL_LIT	3094 #	9395	9399	9420	9430	9449	9459	9558	9569	9690	9770	12384
	12399	12876	13702	14446	15166	16574	16598	16603	17565	20156	20449	20553
	29836	31514	34379	34907	34912	35013	35114	35405	35778	35782	35801	36092
	36248	36619	36626	37958	38084	38205	41210	41228	41336	41428	41642	41663
	41692	41697	41702	47519	47659	47899	48032					
OLITO.SL_LIT	3096 #	9669	20544	20548	35408	35905	35913	36007	36082	36088	36096	36120
	36251	36615	37881	39468	39484	41310	47515	47804	47903			
OLIT16	3112 #	5495	28493	36101	37282	37311	37316	37321	38850	39120	48096	48099
OLIT24	3113 #	5614	5625	17634	19468	19477	19482	24477	25367	36106	36353	37194
	47651											
OLIT8	3111 #	20211	20322	20331	20339	21164	41647	41668				
PL	3090 #	11379	12554	18240	18249	20224	29485	29776	35271	35293	37512	47749
PL.SL_WB	3095 #	18198	18217	18274	18286	18315	18330	18656				
PL_MSS	3084 #	11358	12524	12531	12585	18230	18235	20215	20796	21310	29478	29773
	35034	35298	37509	47719	47731							
RL.MM.P	3053 #	8496	8510	18702	18754	46238	46290					
RL.MM.PTE	3054 #	8150	35621	40074	40079	40157	40162	40244	40319	40336	40420	40425
	40609	40639	40642									
RL.RM.4	3052 #	7655	7667	7895	7919	28558	28573	28725	28745	30913	30918	32274
	32277	32280	32285	33207	35954							
RL.RM.P	3050 #	9735	9786	18723	18750	31591	31595	31598	31603			
RL.RM.PS	3051 #	18734	18761									
RL.RR.P	3055 #	18738	18742									
RR.MM.P	3062 #	18698	18714	29494	29859	29872	29881	29885	29891	29895	34387	35551
	38087	38101	38209	41213	41231	41377	41655	41726				
RR.MM.PS	3063 #	18423										
RR.MM.SIZ	3064 #	6163	6580	6600	6617	6651	6769	6782	6790	7398	7426	7967
	8050	11114	11166	11177	11386	12606	13312	13323	13355	13366	13949	13959
	13968	13977	15351	15362	15366	15371	15375	15476	15519	17587	22751	22761
	22764	24692	24815	24973	25095	25262	26607	27494	27506	27520	27711	27941
	27945	27970	28021	28217	28401	28411	28942	28945	28962	28971	28978	28982
	31175	31178	31860	31914	31932	31947	32155	32445	32466	32485	32501	32963
	32968	33001	33006	33131	33134	33164	33167	33175	33503	33835	34008	34304
	34607	34613	35026	35256	35275	35308	35319	35334	35347	35363	35440	35445
	35450	35579	35951	35976	36015	36034	36146	36299	36635	36639	37491	38822
	39015	39024	39048	39146	39397	41447	41761	41770	41781	42658	46996	47629
	47639	47679	47689	47797	47814	47886	47889	47892				
RR.MR.4	3059 #	6482	7792	7812	17546	18400	27498	27512	27525	27533	30745	30826
	32200	32203	32206	32210	32237	32240	32243	32247	33562	35433	36043	36068
	45056											
RR.MR.9	3061 #	11383	12603	36128								
RR.MR.P	3057 #	11373	12545	12590	12615	13688	15108	15136	15146	15169	15187	15212
	15220	16578	31617	31622	31626	31629	31634	47666	47670	47736	47776	47844
	47848	47851	47873	47877	47881	47988	47991	48036				
RR.MR.PS	3058 #	18430	18486	18729	18757							
RR.MR.S	3060 #	18707	18746	47788	47793							
RR.RR.P	3065 #	18663										

Cross Reference Listing - Field Names and Defined Values

```
RR.RR.PS        3066 #  18482
RR.RR.SIZ       3067 #  6620    7031    16013   16287   16486   17601   19009   19698   22781   23680   23932
                24675   24721   24747   24785   24983   25001   25030   25061   25161   25196   25210   25276
                25288   27502   27674   27768   27937   27989   28014   28151   28703   28846   29100   29103
                29109   29120   29180   29223   29253   29280   29334   29346   29391   29424   29734   31166
                31214   31221   31238   31578   31857   31871   31881   31920   32034   32040   32085   32115
                32139   32143   32692   32888   32893   32906   33155   33715   33731   34309   34462   34501
                34623   34654   34663   34676   34710   34719   34736   34740   34744   35546   35971   36065
                36072   37143   38366   38452   38827   38837   39074   41404   46915   47015
SL              3091 #  9805    18244
SL.PL_WB        3092 #  8492    8506    9665    12221   13466   13615   13698   15102   15130   15205   15886
                18363   18367   18371   18521   18524   18671   18678   19142   19163   35284   39493   41297
                41372   41400   41464   46212   46230   46235   46295   47952
XZ.MM           3041 #  15398   18435   18447   20560   20564   20597   35430   35908   36012   36258   36622
                38039   47613   47619   47695
XZ.MR           3040 #  18442   18454   18501   18516
XZ.PTX          3044 #  40300   40506   40531   40537   40630
XZ.RR           3042 #  18497   18512   36254   36630   39497   39500
XZ.VPN          3043 #  40482   40487   40492   40621
ZERO            3098 #  5352    5362    5370    5375    5388    5392    5413    5417    5431    5438    5441
                5448    5451    5526    5568    5655    5667    5671    5726    5754    5760    5880    5887
                5892    5903    5907    5924    5927    5929    5960    6152    6177    6251    6260    6277
                6590    6647    6678    6685    6743    6746    6748    6756    6760    6772    6806    6811
                6822    6828    6861    6864    6867    6871    6875    6879    6886    6892    6895    6900
                6907    6914    6925    6935    6951    6956    6959    6969    6972    6987    6990    7017
                7021    7027    7116    7119    7122    7125    7160    7180    7192    7198    7207    7215
                7226    7238    7271    7287    7290    7294    7389    7392    7396    7423    7447    7469
                7492    7514    7536    7588    7694    7698    7721    7725    7804    7823    7876    8097
                8103    8121    8127    8131    8135    8139    8146    8153    8163    8167    8173    8278
                8306    8320    8356    8377    8403    8415    8440    8459    8471    8501    8521    8526
                8535    8606    8695    8780    8824    8869    8985    8992    9031    9071    9098    9103
                9309    9339    9376    9387    9415    9424    9494    9498    9514    9518    9522    9526
                9720    9754    9760    9791    9812    10692   10697   10702   10751   10763   10770   10773
                10820   10829   10872   11101   11106   11110   11238   11250   11258   11454   11478   11485
                11667   11675   11686   11696   11703   11774   11787   11896   11907   11913   11928   11933
                12056   12140   12227   12336   12347   12378   12395   12404   12541   12565   12569   12611
                12645   12693   12863   12881   12962   12982   13073   13165   13244   13317   13360   13472
                13571   13576   13581   13595   13609   13629   13640   13663   13673   13771   13796   13866
                13954   13973   13990   14038   14043   14048   14058   14069   14093   14102   14107   14212
                14314   14339   14434   14508   14519   14676   14695   14720   14728   14739   14748   14757
                14762   14767   14777   14782   14789   14794   14802   14807   14814   14819   14850   14866
                14964   14984   15113   15118   15140   15151   15156   15175   15192   15191   15197   15215
                15225   15231   15242   15318   15356   15471   15486   15493   15678   15693   15698   15706
                15729   15736   15746   15770   15792   15868   15892   15909   15985   16005   16029   16047
                16052   16124   16203   16225   16234   16238   16242   16255   16259   16262   16273   16277
                16284   16414   16425   16428   16452   16456   16460   16466   16477   16496   16501   16515
                16520   16538   16543   16588   16628   16633   16637   16649   16722   16733   16761   16791
                16797   16802   16833   16844   16910   16922   16926   16942   16973   17004   17025   17048
                17070   17110   17155   17164   17195   17199   17210   17214   17218   17221   17232   17274
                17278   17294   17311   17316   17332   17441   17448   17454   17460   17466   17470   17477
                17481   17486   17499   17507   17511   17516   17534   17578   17616   17619   17625   17629
                17638   18202   18278   18373   18382   18427   18439   18451   18726   18765   18769   18772
                18970   18975   18980   18985   18990   18995   19000   19004   19348   19524   19533   19539
                19545   19575   19607   19611   19615   19619   19623   19627   19631   19635   19643   19658
                19663   19668   19673   19678   19683   19688   19692   19701   19715   19726   19733   20110
```

Cross Reference Listing - Field Names and Defined Values

20116	20144	20160	20164	20168	20186	20197	20203	20229	20239	20248	20257
20265	20274	20283	20292	20300	20309	20318	20327	20335	20344	20414	20424
20433	20441	20462	20471	20479	20556	20579	20584	20589	20603	20724	20730
20745	20764	20773	21025	21099	21106	21122	21126	21131	21201	21221	21228
21232	21239	21243	21248	21253	21275	21361	21442	21458	21464	21482	21512
21517	21535	21549	21584	21618	21650	21705	21715	21771	21775	21780	21804
21819	21823	21878	21899	21972	21994	22008	22015	22054	22057	22092	22418
22424	22435	22446	22450	22454	22457	22467	22473	22477	22508	22514	22525
22531	22541	22567	22574	22581	22603	22609	22615	22623	22627	22630	22642
22652	22666	22673	22683	22687	22697	22704	22715	22721	22725	22745	22772
22929	22933	22939	22946	22949	22952	22955	22985	23007	23017	23021	23045
23073	23090	23102	23133	23137	23154	23158	23182	23210	23214	23225	23230
23234	23248	23258	23283	23288	23306	23310	23338	23342	23353	23357	23464
23479	23483	23493	23497	23500	23509	23520	23555	23569	23573	23669	23674
23684	23693	23699	23702	23730	23734	23782	23786	23844	23848	23897	23936
23943	23969	23974	23992	24001	24024	24078	24098	24112	24245	24307	24333
24376	24433	24437	24442	24446	24450	24456	24489	24500	24506	24515	24531
24551	24560	24629	24635	24640	24655	24688	24697	24701	24712	24715	24735
24741	24744	24752	24757	24768	24778	24789	24792	24808	24819	24824	24828
24843	24854	24859	24863	24873	24876	24883	24925	24936	24948	24963	24969
24992	24997	25015	25022	25035	25056	25069	25080	25084	25098	25102	25113
25122	25128	25177	25207	25213	25302	25307	25312	25318	25322	25325	25357
25363	25371	25376	25379	25382	25388	25394	25401	25405	25625	25629	25639
25643	25686	25691	25800	25817	25821	25827	25832	25837	25842	25931	25935
25953	25957	26095	26123	26125	26130	26135	26150	26153	26168	26176	26184
26189	26198	26216	26223	26237	26240	26274	26277	26306	26333	26345	26348
26353	26357	26435	26444	26472	26475	26480	26483	26486	26489	26496	26509
26512	26515	26603	26615	26620	26623	26626	26636	26670	26674	26678	26682
26819	27007	27029	27032	27035	27038	27055	27071	27075	27112	27117	27121
27125	27129	27140	27145	27292	27315	27325	27328	27336	27341	27417	27420
27423	27427	27441	27465	27668	27677	27682	27692	27696	27720	27723	27752
27813	27919	27922	27949	27979	28003	28060	28086	28109	28121	28125	28128
28131	28134	28166	28171	28208	28278	28302	28306	28554	28593	28606	28622
28632	28636	28640	28686	28691	28764	28809	28831	28852	29083	29106	29124
29154	29159	29166	29184	29201	29218	29234	29238	29257	29272	29284	29360
29365	29376	29402	29409	29412	29421	29427	29462	29469	29481	29542	29570
29574	29578	29615	29623	29627	29663	29674	29692	29702	29706	29739	29742
29745	29748	29754	29758	29768	29808	29814	29831	29834	29854	29924	29927
29949	29953	29968	29973	29977	29981	29986	29991	29996	30001	30005	30009
30011	30019	30025	30033	30036	30039	30042	30045	30049	30054	30083	30089
30094	30099	30108	30113	30125	30130	30174	30180	30185	30189	30205	30210
30220	30289	30324	30331	30336	30342	30369	30384	30391	30395	30427	30433
30438	30443	30484	30535	30567	30606	30644	30648	30655	30658	30664	30668
30673	30723	30728	30832	30836	30848	30852	30856	30860	30864	30896	30902
30936	30941	30946	30982	30990	31032	31039	31043	31068	31119	31123	31126
31128	31133	31139	31163	31172	31181	31218	31224	31233	31235	31252	31256
31260	31285	31290	31299	31305	31312	31333	31426	31430	31460	31469	31479
31482	31493	31714	31719	31775	31779	31887	31891	31911	31941	32029	32049
32054	32058	32071	32039	32093	32096	32135	32158	32176	32514	32525	32723
32727	32730	32733	32737	32843	32847	32857	32925	32929	33011	33030	33040
33046	33082	33146	33171	33473	33492	33500	33509	33514	33522	33525	33549
33553	33571	33623	33642	33663	33776	33782	33788	33792	33798	33809	33813
33858	33862	33898	33907	33952	34011	34020	34067	34084	34192	34253	34279
34290	34314	34349	34372	34408	34432	34435	34489	34494	34535	34542	34544
34565	34594	34619	34666	34669	34673	34685	34688	34692	34724	34728	34732

Cross Reference Listing - Field Names and Defined Values

34754	34758	34841	34845	34849	34867	34872	34876	34886	34952	34956	34967
34982	35042	35046	35058	35070	35074	35086	35135	35139	35147	35151	35159
35163	35172	35179	35190	35195	35199	35208	35217	35221	35230	35234	35243
35260	35279	35303	35353	35356	35359	35422	35464	35473	35507	35539	35557
35612	35616	35624	35635	35639	35644	35746	35759	35763	35770	35774	35786
35790	35794	35805	35829	35845	35868	35876	35880	35884	35888	35892	35900
35943	35982	35991	35996	36018	36026	36038	36076	36125	36131	36160	36166
36173	36263	36267	36271	36275	36279	36283	36327	36348	36360	36465	36474
36491	36522	36527	36549	36558	36575	36705	36718	36729	36736	36739	36763
36767	36771	36776	36791	36793	36802	36812	36857	36861	36875	36885	36899
36911	36914	36919	36922	36929	36935	36941	36965	37199	37226	37230	37235
37240	37265	37287	37294	37298	37302	37306	37326	37360	37368	37419	37651
37657	37666	37704	37733	37747	37755	37773	37776	37810	37816	37819	37872
37908	37911	37984	37993	37997	38001	38005	38160	38163	38262	38275	38292
38296	38301	38354	38357	38373	38399	38408	38456	38460	38466	38476	38482
38486	38492	38662	39011	39066	39070	39078	39090	39162	39215	39329	39338
39354	39364	39381	39403	39407	39411	39415	39419	39434	39441	39445	39449
39452	39462	39474	39478	39644	39692	39722	39729	39736	39743	39801	39809
39816	39874	39889	39895	39899	39902	39908	39918	39935	39938	39964	39974
39979	39983	39987	39990	39996	40033	40039	40049	40054	40059	40065	40070
40095	40140	40184	40199	40207	40249	40260	40265	40297	40306	40313	40365
40375	40380	40390	40393	40398	40439	40441	40444	40512	40555	40566	40584
40636	40655	40692	40701	40755	40762	40814	40821	40854	40862	40869	40873
40901	40905	40909	40913	40929	40933	40937	40941	40989	40994	40999	41005
41021	41059	41097	41100	41111	41118	41135	41138	41143	41148	41154	41170
41221	41247	41251	41257	41268	41276	41279	41301	41332	41346	41423	41451
41460	41480	41519	41566	41587	41591	41603	41619	41742	41813	41863	41895
41926	41952	42014	42019	42026	42032	42076	42090	42158	42163	42191	42199
42252	42285	42318	42375	42379	42383	42388	42393	42411	42426	42434	42442
42447	42530	42537	42541	42545	42550	42565	42570	42619	42629	42643	42666
42739	42743	42802	42808	42820	42838	42886	42891	42952	42966	42984	43022
43027	43032	43091	43103	43108	43181	43213	44361	44435	44442	44472	44500
44511	44607	44657	44686	44697	44746	44774	44785	44895	44902	44937	44944
45016	45025	45061	45066	45092	45106	45114	45125	45172	45210	45225	45274
45336	45368	45442	45451	45455	45479	45533	45539	45547	45553	45589	45593
45598	45625	45640	45648	45657	45702	45706	45735	45750	45758	45780	45908
45913	45942	45977	45982	46042	46076	46084	46170	46175	46187	46348	46357
46360	46364	46372	46377	46382	46472	46521	46560	46563	46566	46598	46632
46665	46720	46771	46775	46869	46878	46980	47052	47059	47066	47204	47244
47268	47274	47284	47326	47331	47341	47345	47352	47419	47634	47684	47762
47766	47772	47828	47832	47835	47839	47908	47958	47963	47969	47973	47978
48000	48006	48012	48043	48048							
ZLITO	3101 #	5333	5357	5365	5384	5402	5471	5476	5503	5534	5539	5559
	5617	5630	5662	5681	5731	5784	5796	5939	5977	5997	6006	6127
	6148	6155	6159	6170	6174	6224	6228	6239	6255	6264	6269	6280
	6284	6292	6298	6306	6312	6320	6326	6334	6340	6348	6354	6362
	6368	6376	6382	6386	6390	6396	6404	6410	6418	6424	6432	6463
	6470	6477	6486	6502	6507	6527	6535	6541	6544	6556	6612	6752
	6803	6825	6831	6857	6931	7041	7106	7110	7165	7173	7176	7183
	7230	7263	7267	7275	7279	7283	7309	7313	7321	7329	7337	7344
	7352	7360	7368	7376	7382	7404	7409	7419	7432	7437	7453	7458
	7465	7475	7480	7498	7503	7520	7525	7532	7542	7547	7556	7560
	7564	7568	7577	7594	7614	7626	7634	7643	7647	7651	7663	7675
	7689	7705	7713	7717	7729	7740	7745	7749	7753	7756	7761	7768
	7789	7809	7828	7832	7849	7860	7871	7907	7911	7915	7949	7973

Cross Reference Listing - Field Names and Defined Values

7985	8008	8057	8062	8067	8072	8075	8079	8093	8143	8186	8190
8198	8205	9778	10925	10929	10933	11029	11034	11149	11154	11326	11340
11798	12236	12331	12341	12510	12549	12579	12594	12619	12636	12641	12699
12709	12721	12724	12857	13084	13478	13585	13622	13682	13875	14086	14112
14116	14441	14500	14505	14665	14687	14700	14710	14835	14970	15005	15012
15087	15093	15098	15123	15389	15417	15421	15425	15689	15710	15740	15775
15827	15846	15874	15879	15903	15996	16059	16113	16118	16136	16140	16198
16230	16290	16435	16482	16524	16560	16666	16990	17117	17122	17159	17168
17172	17176	17237	17474	17562	17608	19064	19153	19236	19241	19582	20123
20127	20134	20138	20200	20243	20252	20261	20269	20278	20287	20296	20304
20313	20349	20354	20359	20364	20369	20374	20378	20383	20388	20468	20490
20494	20498	20536	20607	20613	20617	20622	20626	20632	20636	20642	20646
20652	20656	20662	20666	20671	20675	20681	20685	20691	20701	20711	20720
20742	20791	20810	20899	20913	20929	20962	20965	21102	21115	21136	21206
21213	21262	21324	21327	21438	21453	21467	21471	21542	21546	21603	21615
21621	21655	21664	21697	21730	21758	21786	21794	21808	21848	21858	21867
21874	21888	21903	21910	21951	21968	21978	21997	22002	22011	22020	22637
22677	22785	23004	23049	23078	23106	23129	23186	23204	23237	23278	23332
23688	23760	23807	23816	23957	24014	24061	24102	24250	24365	24422	24471
24540	24647	24664	24669	24707	24760	24805	24868	24929	24933	25040	25109
25119	25203	25283	25292	25299	25611	25634	25648	25713	25790	25796	25973
25994	26002	26006	26009	26017	26029	26110	26142	26146	26211	26293	26361
26364	26370	26373	26464	26500	26504	26581	26895	27013	27017	27022	27026
27043	27051	27062	27067	27101	27331	27665	27686	27731	27774	27778	27796
27800	27817	27827	27834	27838	27844	27848	27854	27858	27868	27872	27876
27911	27915	27983	28007	28027	28048	28064	28075	28106	28117	28143	28147
28154	28176	28182	28187	28213	28245	28263	28268	28284	28296	28311	28327
28333	28337	28341	28417	28427	28453	28476	28499	28506	28532	28537	28562
28597	28601	28617	28660	28665	28669	28769	28773	28778	28783	28792	28796
28823	28827	28835	28855	29074	29089	29112	29115	29133	29137	29141	29176
29246	29249	29268	29380	29488	29491	29559	29565	29599	29636	29679	29684
29697	29710	29715	29721	29726	29818	29821	29828	29842	29849	29868	29876
29911	30104	30201	30225	30242	30247	30252	30256	30265	30271	30276	30280
30284	30295	30351	30373	30377	30387	30451	30456	30460	30480	30491	30498
30518	30551	30584	30599	30609	30618	30624	30630	30635	30640	30713	30734
30755	30759	30780	30785	30873	30878	30887	30908	30926	30987	30997	31001
31007	31012	31019	31025	31048	31061	31111	31227	31230	31241	31348	31441
31455	31498	31504	31587	31649	31744	32044	32080	32099	32122	32131	32162
32172	32371	32379	32384	32395	32399	32403	32420	32423	32426	32438	32459
32478	32482	32494	32511	32545	32564	32567	32570	32590	32594	32598	32602
32607	32611	32615	32624	32628	32638	32648	32658	32663	32667	32686	32746
32790	32813	32822	32828	32832	32836	32915	32919	33018	33026	33037	33060
33064	33128	33150	33345	33482	33527	33533	33538	33575	33609	33614	33637
33648	33668	33672	33676	33819	33831	33902	33956	33998	34003	34033	34071
34088	34145	34149	34167	34177	34197	34259	34273	34301	34318	34362	34393
34440	34458	34498	34560	34610	34658	34679	34682	34701	34707	34713	34837
34863	34890	34899	34903	35030	35118	35126	35251	35268	35328	35468	35481
35488	35520	35531	35535	35566	35570	35574	35741	35797	35813	36051	36063
36111	36115	36136	36227	36232	36237	36241	36287	36303	36315	36335	36478
36495	36562	36579	36667	36683	36700	36715	36745	36781	36808	36841	36847
36853	36871	36882	37098	37114	37119	37129	37135	37246	37251	37255	37379
37388	37426	37436	37440	37628	37634	37640	37646	37697	37726	37768	37876
38009	38091	38105	38112	38213	38228	38270	38278	38345	38351	38370	38379
38384	38389	38404	38419	38427	38433	38443	38472	38546	38551	38561	38570
38577	38609	38626	38631	38637	38643	38653	38698	38705	38711	38716	38721

Cross Reference Listing - Field Names and Defined Values

	38728	38739	38745	38751	38756	38806	38916	39209	39268	39274	39385	39725
	39739	39759	39805	39813	39833	39948	39959	40086	40090	40149	40153	40168
	40172	40190	40194	40309	40324	40327	40406	40431	40435	40497	40502	40515
	40527	40541	40547	40551	40560	40571	40576	40589	40593	40625	40646	40650
	41030	41068	41366	41672	41683	42052	42056	42060	42086	42175	42486	42661
	42694	42788	42938	43079	43190	44750	45168	45603	45785	46151	46155	46162
	46165	46179	46182	46213	46243	46251	46279	46305	46313	46344	46352	46367
	46386	46391	46402	46407	46416	46427	46432	46441	46452	46457	46475	46478
	46487	46492	46543	46569	46576	46583	46590	46595	46601	46608	46615	46622
	46629	46635	46642	46648	46655	46662	46668	46675	46682	46689	46696	46716
	46724	46729	46739	46744	46747	46763	46780	46807	46810	46814	46818	46827
	46831	46886	47	46895	46911	46922	46925	46947	46951	46993	47003	47012
	47022	47062		47090	47094	47097	47101	47108	47112	47139	47153	47165
	47176	47197		47222	47237	47251	47262	47293	47298	47302	47336	47356
	47369	47378	93	47405	47411	47422	47469	47473	47477	47493	47508	47539
	47544	47549	559	47573	47780	47808	47947	47982	48084	48087	48090	48093
ZLIT12	3104 #	7065	077	7089	7101	10816	10825	10867	10878	11768	11780	11885
	12033	12040	12371	12516	12703	12715	12729	12734	13670	14424	14840	16063
	17592	20175	20778	20786	47498	47503						
ZLIT16	3105 #	5327	5488	5499	5507	5715	5823	5898	5912	5951	5954	5957
	5963	5966	6132	6221	6232	6243	6247	6660	6665	6707	6964	6977
	7048	7943	7958	7962	8013	8024	8030	20972	20975	23030	23034	23038
	23095	23167	23171	23175	24952	25243	29589	31926	32644	32706	32937	32952
	34405	35062	35066	35078	35082	35090	35185	35314	35368	35375	35381	35385
	35390	35393	35402	35411	35417	35436	35459	35477	35492	35817	35821	35825
	35833	35837	35841	35986	35999	36004	36046	36904	36908	37104	37204	37374
	37384	37400	37536	37559	37563	37585	38064	38095	38109	38156	38173	38202
	38217	38222	38225	38855	39038	41354	41651	41678	47430			
ZLIT20	3106 #											
ZLIT24	3107 #	5519	5523	5605	5917	5933	6142	6235	7013	7036	7094	7169
	7187	7234	16069	20539	20750	20954	25676	25680	26789	26793	26847	26852
	27374	27378	29511	29534	29538	29584	29604	29659	29669	31688	31703	31707
	32573	32654	32670	32861	32878	32883	32903	32983	32991	32996	33051	34569
	35106	35110	35122	35176	35204	35372	35497	35502	35511	35525	35588	35669
	35849	35852	35856	35860	35872	35896	35918	35922	35926	35930	35934	35938
	36060	36139	36183	36732	36755	36799	36916	37109	37270	37274	37278	37431
	37661	37671	37693	37718	37736	37743	37801	37807	37813	37830	37839	37844
	37853	37856	37914	37961	37964	37968	37972	37976	38152	38282	39375	39430
	39923	39930	40179	40217	40232	40241	40303	40509	42673	46148	47157	47161
	47168	47172	47181	47187	47190	47200	47207	47214	47218	47227	47230	47240
	47247	47254	47258	47265	47271	47281	47401	47415	47646			
ZLIT28	3108 #	5515	5600	5687	5723	6273	20428	20437	24462	24525	35331	37722
	37787	40225	40237	40597	40605	47135	47654					
ZLIT4	3102 #	5454	5458	5462	15083	20393	20398	20403	20408	20695	20705	20715
	35515											
ZLIT8	3103 #	5398	5408	5530	5590	5659	5675	5712	5749	5943	5991	7051
	7054	7058	7062	7070	7074	7082	7086	7098	7372	7581	7591	11811
	12559	14733	15071	15077	20453	20572	21384	21394	22428	22481	22485	22545
	22999	23245	23437	23922	23946	24494	24680	24683	24782	24796	24801	24834
	24959	25025	25045	25050	25090	25165	29594	32634	32701	33541	33545	33578
	33618	33629	33653	33659	33916	34028	34075	34092	34132	34153	34157	34171
	34240	34263	34267	34294	34504	35455	35584	35598	35696	35701	36151	36156
	36678	37685	37867	38766	38927	39157	42113	45673	46201	46874	47740	47757
	47935	47942										
ZLITPL	3109 #	19041	19052	19076	19166	20804	21318	35287	37486			

ROTSRK	3142 #					
ABSVAL.140	3147 #					
ABSVAL.141	3148 #					
ABSVAL.142	3149 #					
ABSVAL.143	3150 #					
ABSVAL.150	3151 #					
ABSVAL.151	3152 #					
ABSVAL.152	3153 #					
ABSVAL.153	3154 #					
ABSVAL.160	3155 #					
ABSVAL.161	3156 #					
ABSVAL.162	3157 #					
ABSVAL.163.D	3144 #					
ABSVAL.170	3158 #					
ABSVAL.171.D	3145 #					
ABSVAL.172	3159 #					
ABSVAL.173.D	3146 #					
ASCIISIGN.050	3161 #					
ASCIISIGN.051	3162 #					
ASCIISIGN.052	3163 #					
ASCIISIGN.053	3164 #					
ASCIISIGN.070	3165 #					
ASCIISIGN.071	3166 #					
ASCIISIGN.072	3167 #					
ASCIISIGN.073	3168 #	30454				
BCDSIGN.040	3184 #					
BCDSIGN.041	3185 #					
BCDSIGN.042	3186 #					
BCDSIGN.043	3187 #					
BCDSIGN.060	3188 #					
BCDSIGN.061	3189 #					
BCDSIGN.062	3190 #					
BCDSIGN.063	3191 #	30356				
DSIZE.020	3170 #					
DSIZE.021	3171 #					
DSIZE.022	3172 #					
DSIZE.023	3173 #					
DSIZE.030	3174 #					
DSIZE.031	3175 #					
DSIZE.032	3176 #					
DSIZE.033	3177 #					
PL.EQ.0.123	3182 #					
PL.EQ.0.SIGN.120	3179 #	9511				
PL.EQ.0.SIGN.121	3180 #	41306	41409			
PL.EQ.0.SIGN.122	3181 #					
PL5.003	3214 #					
PL5.013	3215 #					
SL.EQ.0.100	3204 #					
SL.EQ.0.SIGN.101	3202 #					
SL.EQ.0.SIGN.102	3203 #					
VIELD.000	3192 #	18410	18414	18418	18695	41315
VIELD.001	3193 #					
VIELD.002	3194 #					
VIELD.010	3195 #					
VIELD.011	3196 #					

VIELD.012	3197 #	18460	18465	18470	18664							
VIELD.110	3198 #											
VIELD.111	3199 #											
VIELD.112	3200 #											
WBRANGE.130	3208 #											
WBRANGE.131.D	3206 #											
WBRANGE.132	3209 #											
WBRANGE.133.D	3207 #											
WX.NE.0.103	3212 #											
WX.NE.0.113.D	3211 #											
RSRC	3218 #											
DST.R	3277 #	8350	8356	8403	8435	8440	8521	8526	8595	8619	8684	8708
	8768	8812	8856	8913	8930	8941	8985	9043	9048	9318	9327	9335
	9339	9514	9518	9522	9526	9686	9720	9725	9730	9741	9747	9754
	9760	9773	10692	10697	10702	10751	10763	10773	10820	10829	11258	11478
	11485	11667	11675	11686	11696	11703	11774	11787	11896	11907	11913	13165
	13244	13771	13796	14314	14339	14720	14748	14814	14819	15318	15486	16740
	16753	16767	16808	16823	16865	16874	16948	16954	18240	18244	18278	18290
	19448	21035	21040	21361	21549	21804	21899	30001	40989	40999	41005	41024
	41035	41042	41062	41073	41080	41100	41111	41118	41231	41257	41268	41279
	41566	41571	41599	41607	41626	41655	41761	41770	41819	41829	41835	41858
	41890	42209	42220	42619	42629							
DST.R+1	3278 #	8415	8535	9071	9763	9781	9834	9839	11928	11933	14802	14807
	15493	16780	16892									
DST.ROR1	3279 #											
ESP	3256 #	36275	36763	36929	37298	39407	39445	46638				
GPR.R	3274 #	6806	6925	8244	8265	8306	8367	8377	8510	8552	8589	8635
	8678	8726	8762	8806	8850	9505	14777	14782	16722	16733	16761	16797
	16833	16844	16937	18482	18486	18497	18501	18512	18516	18663	18738	18742
	18746	18750	18765	18769	19173	19177	19199	19203	19207	19211	19215	19219
	19335	19341	19352	19357	19394	19398	19455	19557	19575	19593	20773	21275
	36705	36857	38970	38974	39497	40986	41021	41059	41097	41135	41171	41174
	41184	41221	41247	41252	41301	41321	41327	41332	41404	41447	41481	41485
	41493	41520	41524	41532	41587	41595	41688	41708	41712	41742	41749	41814
	41921	41952	42014	42026	42032	42076	42147	42188	42191	42199	42225	42247
	42280	42313	42375	42383	42388	42434	42442	42447	42477	42537	42545	42615
	42735	42743	42752	42757	42762	42773	42776	42782	42784	42792	42802	42803
	42829	42838	42845	42864	42881	42891	42900	42905	42910	42921	42925	42931
	42934	42942	42952	42954	42975	42984	42991	43005	43032	43041	43046	43051
	43062	43066	43072	43075	43083	43091	43093	43117	43125	43130	43139	43157
	43166	43197	43207	44348	44352	44367	44377	44386	44396	44401	44409	44424
	44454	44472	44483	44492	44500	44507	44511	44527	44544	44567	44578	44588
	44596	44603	44607	44623	44640	44657	44668	44678	44686	44693	44697	44717
	44729	44746	44757	44766	44774	44781	44785	44819	44824	44835	44845	44855
	44864	44869	44877	44889	44913	44932	44954	44963	44972	44977	44985	45010
	45031	45038	45061	45073	45083	45092	45101	45106	45114	45136	45147	45172
	45185	45199	45210	45220	45225	45233	45245	45279	45290	45300	45308	45318
	45323	45331	45354	45359	45373	45383	45392	45402	45407	45415	45442	45446
	45460	45470	45479	45489	45494	45502	45523	45564	45589	45593	45607	45616
	45625	45635	45640	45648	45702	45706	45717	45726	45735	45745	45750	45758
	45772	45800	45821	45832	45842	45852	45861	45866	45874	45898	45903	45919
	45931	45942	45953	45959	45967	46008	46012	46023	46033	46042	46052	46057
	46065	46170	46175	46357	46377	46419	46423	46435	46533	46771		
GPR.R+1	3275 #	16791	16802	18459	18464	18469	18757	18772	39500	45515	45657	45977
GPR.ROR1	3276 #	41408	41451	42019	42037	42090	42158	42163				

Cross Reference Listing - Field Names and Defined Values

```
IPR.R           3280 #  34872   34977   35179   35221   36293   46360   46382   46444   46448   46460   46538
IPR.R+1         3281 #
IPR.ROR1        3282 #
ISP             3259 #  36793   36919   37287   38275   39419   46651
KSP             3255 #  36271   36899   36914   37265   37294   38262   39403   39441   46632
LONLIT          3284 #   5438    5448    5611    5667    5892    6839    7027    7045   21358   29083   30567
                31128   35168   35239   35967   36722   36748   37352   39346   39372   46521   46563   46980
                46983   47008   47052   47059   47326   47341
MM.TEMP1        3236 #  20160   20556   21122   21221   21239   27071   27112   27136   27177   27182   27186
                27258   27264   27298   27303   36474   36491   36558   36575   38362   38366   39692   39801
                39874   39908   39954   39964   39996   40049   40059   40065   40070   40260   40393   40692
                40705   40713   40718   40722   40755   40766   40774   40778   40781   40814   40873   46466
MM.TEMP2        3261 #   6756    8153   27420   27435   27465   27506   27512   27516   27520   27525   27529
                38354   38373   38456   38460   39889   39895   39899   39902   39918   39935   39938   39974
                39979   39983   39987   39990   40033   40039   40265   40375   40380   40444   40901   40905
                40909   40913   40929   40933   40937   40941
MM.TEMP3        3262 #   8121    8127    8131    8135    8139   21228   26216   31068   38357   38393   38437
                38492   39162   39809   40054   40095   40140   40184   40207   40398   40439   40701   40762
                40821   40862
MM.TEMP4        3270 #  39722   39729   39736   39743   39816   46555
MM.TEMP5        3235 #  27417   27431   27441   37747   37773   38449   38452   39943   40074   40079   40157
                40162   40244   40249   40313   40333   40336   40420   40425   40441   40566   40584   40609
                40636   40639   40642   40655
POBR            3263 #  34990   35135   35884   36718   40531   46658
POLR            3264 #  36736   40482
P1BR            3265 #  34999   36739   40537
P1LR            3266 #  36759   40487
PCBB            3260 #  35017   35770   35892   36965
R0              3238 #  15693   15729   15736   15892   15909   16225   16496   16538   17100   17199   17218
                17477   17521   17534   17595   17629   20229   22467   22473   22477   22630   22683   22687
                22725   23056   23085   23110   23137   23141   23144   23190   23214   23230   23234   23253
                23261   23288   23292   23295   23319   23338   23342   23353   23483   23487   23514   23530
                23535   23693   23702   23725   23821   23829   23880   23903   23943   23997   24004   24019
                24027   24092   24098   24107   24115   24189   24215   24219   24238   24307   24323   24376
                24427   24446   24515   24546   24560   24575   24644   24650   24675   24688   24697   24718
                24721   24789   24792   24979   25056   25061   25080   25084   25161   25168   25171   25196
                25207   25210   25213   25249   25262   25276   25288   25302   25312   25325   25953   25957
                26176   26184   26306   26333   26353   26357   26607   27668   27671   27674   27768   27970
                28021   28401   28846   28959   28971   28982   29109   29120   29376   29391   29739   29991
                29996   30009   30019   30039   31119   31172   31181   31235   31853   31857   31860   31864
                31901   31920   31923   31929   31932   31947   32029   32034   32040   32085   32096   32692
                33131   33244   33522   33813   34008   34011   34161   34349   34354   34358   34408   34535
                34542   34688   39078   39082   39090   46665
R1              3239 #   5441    7119   15698   15746   15770   16234   16501   16543   17004   17025   17048
                17195   17221   17253   17299   17460   17516   17550   20239   22457   22627   22673   22721
                22985   23210   23526   23674   23684   23875   23936   23974   24010   24024   24245   24327
                24333   24433   24489   24506   24531   24565   24569   24704   24873   24880   25177   25182
                25376   25388   25394   25620   25660   25806   26123   26198   26240   26283   26341   26472
                26593   27704   29213   29355   29742   29981   29986   30488   30521   31126   31233   31469
                31527   31905   32111   33514   33518   34368   34550   34685   34724   34728   34732   39117
                46671
R10             3248 #  20318
R11             3249 #  20327
R12             3250 #   5370    5375   20335   20441   20471   20475   20584
R13             3251 #  20344   20433   20462   20530   20589
```

Cross Reference Listing - Field Names and Defined Values

R2	3240 #	5451	7122	15916	16005	16010	16013	16047	16073	16203	16211	16262
	16267	16287	16510	16548	17030	17052	17303	17481	17491	17511	17587	17601
	17619	17625	17638	20248	22435	22623	22642	22715	22751	22781	22801	22804
	22933	23509	23680	23699	23932	24450	24557	24632	24640	24747	24757	24948
	24956	24983	24997	25001	25015	25030	25064	25102	25307	25357	25371	25379
	25725	25931	25935	26345	26871	26880	27007	27720	27747	27884	27919	28248
	28942	29092	29097	29100	29103	29106	29166	29218	29360	29402	29745	29927
	30005	30025	30042	30049	30108	30113	30384	30391	30395	30655	30658	30728
	31163	31166	31169	31175	31178	31214	31218	31221	31224	31238	31740	31748
	31775	31779	31874	31884	31911	31914	31938	32054	32058	32071	32089	32115
	32139	32143	32719	32723	33137	33164	33167	33175	33549	33715	33731	33782
	33788	33792	33798	33835	33858	33898	33952	34067	34084	34192	34309	34314
	34544	34607	34613	34654	34663	34666	34669	39002	39006	39028	39053	39060
	39066	39070	39096	46678								
R3	3241 #	5431	7113	15706	16052	16124	16128	16277	16284	16520	17110	17155
	17164	17470	17507	17616	20257	22446	22772	22916	22949	22955	22972	22988
	22991	23073	23115	23119	23196	23201	23248	23265	23269	23324	23329	23500
	23540	23544	23555	23669	23825	23833	23838	23884	23888	24078	24088	24112
	24195	24442	24500	24551	24572	24701	24715	24808	24819	24828	24831	24854
	24883	24963	25113	25122	25128	25322	25363	25625	25629	25691	25717	25868
	26125	26189	26223	26255	26277	26337	26475	26645	26670	26678	26876	27048
	27723	28407	28974	29238	29338	29754	29758	29953	29968	29973	29977	30011
	30045	30054	30359	30363	30400	31133	31139	31248	31252	31256	31260	31479
	31719	31752	31908	32118	33556	34553	34594	34599	34673	39011		
R4	3242 #	16238	16460	16466	16477	16486	16633	17210	17274	17294	20265	22929
	23497	24672	24678	24763	24778	24785	26512	27711	28217	28397	28411	28852
	28945	28956	28962	28978	29086	29159	29180	29223	29253	29272	29280	29334
	29346	33134	34565	34619	34623	34676	34692	34710	34719	34736	34740	34744
	39103	46685										
R5	3243 #	5352	5362	5413	6251	7017	7125	16242	16452	16649	17214	17278
	20274	22946	22952	23493	24629	24735	26480	26483	26486	26489	26674	26682
	26899	27341	28208	28394	28953	29162	29275	33525	33805	33893	33945	33948
	33959	34432	34446	34450	34489	34509						
R6	3244 #	20283	37651	42402	42411	42426	42530	42557	42579	42597	42603	42643
	42652	42715										
R7	3245 #	20292	41926	41930								
R8	3246 #	20300	46692									
R9	3247 #	20309										
RTMPGPR	3253 #	6138	6436	6656	6701	6956	6959	6969	7116	7129	29201	29208
	46550	46775										
SBR	3267 #	35008	40300	40506	40630							
SLR	3268 #	40492	40621									
SP	3252 #	5417	5655	8455	8467	19639	19722	19733	19738	20110	20116	20190
	20195	20203	20418	20444	20503	20508	20513	20730	20734	20738	20745	20764
	20767	21099	21111	21147	21152	21158	21201	21216	21232	21253	21272	28109
	28121	28166	28858	29365	34876	34960	36267	36331	36341	36357	36791	36802
	36817	36861	36878	36891	36911	36922	37199	37230	37235	37240	37326	37330
	37356	37364	37980	37989	38296	38301	38566	38658	38775	39166	39184	39329
	39335	39341	39381	39462	39474							
SPNICR.SPICR	3269 #	35353	35356	35359	35971	38827	38832	38837	38844			
SSP	3257 #	36279	36767	36935	37302	39411	39449	46645				
TEMP.R	3271 #	8163	27941	27945	28060	29692	29702	29706	29734	32445	32466	32485
	32501	32514	32525	32888	32893	32906	46348	46372	46394	46398	46410	46528
	46735											
TEMP.R+1	3272 #											

Cross Reference Listing - Field Names and Defined Values

TEMP.ROR1	3273 #											
TEMP0	3221 #	5546	5552	5776	5806	8146	9111	12347	12389	12693	12849	12982
	13069	13590	13602	13635	13688	13706	13881	14241	14512	14991	15781	16293
	16491	17499	17539	17546	18702	18970	18975	18980	18985	18990	18995	19000
	19004	19009	19607	19611	19615	19619	19623	19627	19631	19635	19658	19663
	19668	19673	19678	19683	19688	19692	19698	19701	19715	20103	21126	21131
	21243	21248	21330	22098	22788	27160	27164	27168	27172	27446	28134	29409
	29412	29578	29615	29688	29730	29854	29899	30355	30923	30957	30962	30972
	30977	32200	33562	33571	33632	33642	33663	33776	36125	36128	36131	36254
	37678	37862	38761	38922	39152	39507	41433	41438	41457	41460	41603	41746
	41755	41766	42252	42257	42285	42290	42318	42322	42511	42515	42541	42658
	43103	44522	44533	44592	44618	44629	44708	44713	44804	45021	45066	45125
	45677	45908	46598	46882	46907	46989	46996	46999	47015	47952		
TEMP1	3222 #	6590	6620	6746	6760	6879	7031	7812	8496	8601	8690	8775
	8819	8864	8981	8989	8992	8995	9098	9387	9735	9786	10770	10872
	11139	11803	11816	12336	12378	12404	13571	13576	13629	13645	14038	14043
	14716	14744	14757	14789	14794	14857	15231	15471	16029	16255	16414	16428
	16628	16637	16915	16922	16942	16978	17448	18400	19047	19060	19070	19148
	19309	19314	19533	19539	19545	21267	21389	21399	21535	21553	21557	21768
	21972	22008	22418	22461	22603	22697	24519	24992	25098	25253	25601	25639
	25643	26088	26150	26237	26417	26509	27140	27206	27216	27226	27449	28131
	28558	28573	28686	28691	28703	29545	29549	29554	30083	30174	30324	30427
	30589	30840	31421	31460	32093	32203	32375	33473	33492	33509	34304	34867
	34952	34956	35433	35546	35621	35624	35746	36729	41718	41722	42379	45056
	45451	45533	45547	45598	45669	45780	46604	46889				
TEMP10	3231 #	6743	6779	6787	6795	6817	6864	6889	6903	6910	6917	6922
	7733	7737	8100	14762	14866	14872	16910	16926	16973	21015	21049	25800
	25884	25890	26603	26699	26704	26709	26714	26730	26735	26740	26745	27029
	27129	28484	28524	32733	37908	37997						
TEMP11	3232 #	25817	25821	25895	25900	26615	26620	26757	26808	27032	27125	28488
	28527	32737	37911	38001								
TEMP12	3233 #	6678	6694	6822	6886	6935	6951	23017	23021	23045	23090	23102
	23154	23158	23182	23464	23520	25827	25832	25906	25912	26623	26760	26811
	27035	27121	27737	27787	27791	27813	27820	27959	27992	28017	28030	28078
	28086	35650	35659	37917	37984	47908						
TEMP13	3234 #	5960	6145	6277	6987	23007	23550	23730	23734	23766	23782	23786
	23811	23844	23848	24712	24859	25837	25842	25919	25926	26626	26763	26814
	27033	27117	27922	27979	27989	28003	28014	28069	28151	33146	33155	35579
TEMP2	3223 #	5760	9031	9034	9058	9066	9103	9115	9376	9697	9816	9820
	9825	11010	11019	11101	11106	11110	11129	11238	11319	11458	11633	11650
	12056	12140	12227	12352	12358	12410	12541	12565	12569	12611	12881	12962
	13073	13079	13472	13581	13595	13651	13657	13866	14233	14253	14434	14460
	14478	14519	14645	14649	14670	14683	14695	14739	14850	14964	15108	15113
	15151	15161	15175	15182	15242	15678	17311	17316	17321	17326	17527	18202
	18207	18226	18320	18423	18427	18430	18435	18439	18442	18447	18451	18454
	18714	18723	20579	21106	21442	21475	21512	21597	21612	21775	21780	21782
	21813	21819	21823	21878	21881	21906	21983	22024	22050	22424	22508	22567
	22609	22704	22969	24536	25401	25405	25872	26095	26435	26819	27086	27502
	27533	27537	28128	29643	29902	29930	29934	30089	30180	30331	30433	31426
	32206	33496	33553	34604	34616	34661	35094	35337	35427	35430	35774	35790
	35908	35954	35957	35982	36142	36160	36173	36258	36263	36348	36622	40869
	46611	47762	47766	47772	47788	47828	47958	47973	47988	48000	48006	48012
	48027											
TEMP3	3224 #	7792	9121	9124	9269	9274	9279	9285	9290	9295	9481	9489
	9552	9563	9579	9589	9594	9599	9606	9612	9617	9623	9628	9633

	9640	9646	9651	9665	11114	11169	11180	11250	11454	12590	12603	12606
	12645	13312	13317	13323	13330	13336	13342	13346	13609	13640	13663	13673
	13949	13954	13959	13973	13990	14052	14062	14073	14093	14207	14218	14258
	14455	14470	14474	14952	15136	15169	15187	15191	15197	16578	17232	17332
	17441	18221	18698	18707	18726	18729	18734	19513	19518	19553	19588	21449
	21531	21539	22525	22536	24815	24876	26099	26104	26130	26274	26439	26451
	26515	26827	27292	28125	28713	28725	28733	28745	30094	30185	30342	30509
	31415	31435	31482	31634	31941	31944	32210	32237	32280	33171	33500	34754
	34758	36295	36465	36469	36486	36549	36553	36570	41314	42081	42143	42153
	47666	47736	47793	47832	47881	47991	48036	48048				
TEMP4	3225 #	7638	7655	7667	7686	9485	9494	9510	11445	11464	11561	13355
	13360	13366	13371	13374	14508	15714	15922	15985	16273	16528	16532	16994
	17070	17242	17454	18395	18405	19524	19528	20183	20186	20219	20233	20518
	21458	21517	21584	21588	21650	21662	21723	21762	21852	21864	21919	22745
	22748	23897	23907	24863	26444	26835	27315	27692	28278	30099	30189	30336
	30438	30574	30829	31039	31043	31108	31591	31629	32240	32277	33073	33503
	36012	36043	36068	36072	41412	41418	46618	47670	47835	47844	47877	
TEMP5	3226 #	7198	7207	7271	7781	7786	7899	7978	8087	11550	11639	11760
	11807	11824	11880	11888	12051	12127	12395	12575	12949	13151	13227	13448
	13761	13780	14304	14323	14935	15118	15140	15156	15215	15225	15799	15850
	16039	16555	16567	16593	18754	18761	20144	20164	20168	20479	20560	20564
	20814	20819	21673	21686	21712	22514	22574	22615	22652	23468	23978	23984
	23988	24039	24047	24073	24082	24128	24135	24225	24231	24256	24262	24313
	24342	24456	26423	26428	26838	27325	27677	27696	29529	29831	29834	30579
	30826	31244	31595	31626	32243	32274	34253	34279	34290	34372	34387	34462
	34501	36018	36026	36056	36327	37143	37226	37479	37482	38292	47396	47613
	47839	47848	47873									
TEMP6	3227 #	6482	8897	8903	9168	9173	9178	9183	9189	9194	9198	9202
	9207	9212	9217	9222	9228	9233	9237	9241	13968	13977	13981	13986
	14102	14984	16259	16456	18659	18692	20597	21025	24944	24969	26135	26192
	26252	26841	27328	27682	28240	28302	28306	28347	29424	29519	29859	29872
	29881	29885	29891	29895	29906	30205	30220	30484	30495	30595	30644	30664
	30668	30673	30679	30699	30723	30731	30745	30771	30896	30902	30913	30918
	30936	30941	30946	30982	30990	31032	31336	31598	31617	31622	32247	33077
	33623	33862	33907	33992	36030	36065	41947					
TEMP7	3228 #	6152	6163	6800	6828	6871	6875	7290	7895	7919	8150	9498
	11982	12221	13460	13466	13615	14212	15308	15786	15886	16025	16654	17466
	17486	17578	17581	17613	20414	21020	22581	23449	23453	26168	26171	26280
	27423	27494	27498	27540	27752	27937	27949	28721	29649	29654	29808	30443
	31333	31678	32794	33115	33119	33207	34696	36307	36311	36630	38163	38167
	46869	46915	46918	47619								
TEMP8	3229 #	6748	6867	7396	7423	7447	7469	8097	12838	12843	13056	13061
	13861	13870	14418	14429	14655	14825	14959	14976	15356	15476	15519	15752
	15816	15822	15868	16246	16419	16472	26153	26496	26636	26641	27336	27356
	27360	28457	28461	31123	31285	31290	31299	31305	31411	31493	31871	31887
	31891	32151	32176	32727	39043	39074	39338	39434	39478	46215	46226	46246
	46560	46625	47695									
TEMP9	3230 #	6177	6772	6811	6814	6843	6895	7287	7294	7492	7514	7536
	7725	8103	21010	21045	27075	27097	27145	27271	27275	28469	28522	29457
	29462	29748	31430	31578	31881	32158	32730	37872	37993	46187	46295	
USP	3258 #	36283	36771	36941	37306	39415	39452					
ZERO	3285 #	5421	5526	5568	5573	5578	5586	5595	5635	5646	5726	5754
	5880	5887	5903	5907	5924	6122	6260	6585	6603	6764	6883	7190
	7192	7203	7215	7223	7238	7392	7658	7670	7799	7804	7818	7823
	7853	8177	8291	8332	8362	8373	8387	8420	8492	8506	8640	8909

Cross Reference Listing - Field Names and Defined Values

8919 8926 8936 8946 9018 9022 9027 9038 9147 9152 9157 9162
9300 9305 9309 9314 9323 9381 9405 9409 9435 9440 9445 9453
9464 9470 9476 9501 9573 9584 9706 10679 10732 10804 10991 10995
11001 11015 11135 11145 11157 11233 11243 11334 11344 11351 11358 11367
11373 11383 11440 11470 11555 11570 11629 11644 11656 11662 11756 11793
11977 12046 12061 12069 12135 12145 12154 12160 12165 12232 12368 12524
12531 12545 12585 12598 12615 12626 12738 12741 12744 12833 12957 12967
12977 12988 13052 13089 13156 13232 13453 13618 13677 13698 13766 13785
14076 14082 14097 14228 14247 14309 14328 14450 14493 14639 14705 14772
14940 14947 15000 15146 15212 15220 15238 15299 15304 15340 15406 15805
15836 15841 15855 15860 15991 16425 16440 16447 16505 16581 16986 17075
18198 18217 18230 18235 18274 18286 18315 18330 18363 18367 18371 18378
18382 18521 18524 18651 18656 18671 18678 18685 18688 18711 18718 18914
19021 19096 19142 19163 19187 19248 19283 19302 19321 19329 19381 19409
19413 19417 19444 19460 19464 19473 19492 19564 19571 19578 19719 20098
20207 20215 20224 20458 20486 20594 20756 20796 21005 21030 21281 21310
21434 21521 21600 21607 21624 21668 21678 21681 21693 21701 21719 21726
21733 21790 21801 21870 21892 21896 21916 21988 22046 22496 22502 22556
22561 22596 22661 22710 22730 22994 23053 23060 23064 23562 23928 23992
24001 24317 24371 24437 24847 24988 25608 25858 25863 26180 26302 26317
26321 26578 26588 27195 27229 27233 27288 27454 27700 28038 28090 28159
28257 28322 28510 28567 28582 28588 28717 28729 28737 29129 29145 29349
29415 29469 29473 29478 29481 29499 29514 29524 29623 29627 29639 29773
29776 29779 29814 29919 29924 29937 30118 30193 30210 30215 30229 30447
30454 30468 30505 30708 30741 30768 30883 30892 31508 31517 31533 31538
31553 31603 31682 31685 31691 32063 32067 32285 32442 32463 32498 32517
32535 32550 32554 32586 32682 32867 32870 33487 33506 34547 34556 34964
34974 35034 35098 35102 35130 35143 35247 35265 35284 35298 35342 35398
35593 35607 35612 35629 35644 35655 35991 35996 36076 36482 36499 36503
36508 36517 36522 36566 36583 36587 36643 36687 36696 36785 36865 36888
37396 37413 37422 37509 37733 37763 37810 37816 38267 38289 38770 39086
39179 39189 39222 39427 39493 39755 39884 40199 40212 40221 40255 40321
40365 40368 40555 40877 40881 40885 40994 41106 41191 41206 41225 41263
41297 41305 41372 41400 41464 41489 41499 41504 41528 41538 41543 41612
41775 41785 41801 41805 41824 41957 41962 41969 41973 41979 41984 41990
41995 42002 42007 42048 42097 42102 42107 42118 42124 42133 42195 42356
42397 42438 42464 42489 42611 42624 42655 42706 42719 42748 42767 42797
42814 42854 42859 42896 42916 42947 42958 43000 43037 43057 43087 43097
43147 43152 43177 43185 43202 43225 43230 43235 44357 44372 44382 44391
44448 44476 44488 44496 44504 44571 44583 44600 44661 44673 44682 44690
44762 44770 44778 44840 44850 44859 44908 44949 44959 44968 44991 44996
45078 45088 45096 45120 45192 45205 45215 45251 45295 45305 45313 45364
45378 45388 45397 45465 45475 45484 45529 45558 45611 45621 45630 45682
45721 45731 45740 45794 45817 45837 45847 45856 45925 45936 45947 45987
46028 46038 46047 46212 46230 46364 46495 46822 46838 47018 47204 47244
47331 47359 47381 47719 47731 47744 47776 47851 47856 47859 47862 47865
48017 48021

ZERO.CLRR6SP 3286 # 38952

SPW 3288 #

MLONG 3292 # 5327 5333 5365 5384 5398 5402 5408 5438 5448 5471 5476
5491 5503 5515 5526 5530 5534 5539 5590 5600 5617 5635 5662
5681 5687 5698 5712 5723 5731 5754 5784 5903 5939 5973 5977
6127 6135 6148 6155 6159 6166 6174 6224 6255 6260 6264 6273
6288 6292 6302 6306 6316 6320 6330 6334 6344 6348 6358 6362
6372 6376 6386 6390 6400 6404 6414 6418 6432 6486 6502 6512

Cross Reference Listing - Field Names and Defined Values

28048	28064	28106	28117	28125	28128	28131	28134	28143	28147	28151	28182
28187	28213	28240	28245	28248	28284	28302	28306	28311	28322	28327	28341
28347	28407	28417	28427	28453	28457	28461	28469	28484	28488	28493	28506
28510	28522	28524	28527	28532	28537	28582	28588	28601	28669	28703	28717
28729	28737	28741	28745	28749	28769	28778	28783	28846	28855	28953	28974
29074	29083	29089	29100	29103	29106	29109	29112	29120	29129	29133	29137
29141	29145	29154	29159	29176	29180	29184	29223	29234	29246	29249	29253
29257	29268	29272	29280	29284	29334	29346	29391	29412	29421	29424	29427
29462	29469	29473	29481	29485	29491	29494	29499	29511	29514	29519	29524
29529	29534	29538	29559	29584	29589	29594	29599	29604	29623	29627	29639
29765	29776	29779	29814	29818	29834	29842	29846	29849	29872	29881	29891
29895	29899	29902	29906	29911	29924	29937	30029	30036	30104	30193	30201
30210	30242	30256	30295	30351	30451	30456	30460	30480	30491	30505	30518
30567	30599	30741	30785	30826	30913	30918	31039	31043	31068	31128	31172
31214	31218	31221	31227	31230	31233	31235	31238	31241	31309	31441	31460
31482	31493	31504	31508	31533	31538	31548	31553	31574	31578	31591	31595
31598	31603	31678	31682	31685	31688	31691	31703	31707	31871	31881	31887
31905	31908	31926	32034	32040	32044	32054	32058	32071	32080	32089	32096
32099	32111	32115	32118	32122	32131	32139	32143	32155	32162	32172	32197
32214	32234	32251	32270	32274	32277	32280	32285	32288	32420	32423	32426
32438	32459	32478	32490	32494	32545	32564	32567	32570	32573	32590	32594
32598	32602	32607	32611	32615	32624	32648	32658	32663	32667	32692	32746
32903	32952	32983	33001	33006	33014	33037	33073	33119	33128	33150	33155
33182	33190	33198	33215	33218	33221	33229	33232	33235	33482	33487	33492
33527	33533	33538	33541	33545	33559	33578	33618	33629	33632	33653	33659
33663	33668	33672	33676	33715	33720	33726	33731	33737	33813	33858	33862
33898	33902	33907	33916	33992	33998	34003	34028	34036	34067	34071	34075
34084	34088	34092	34132	34149	34153	34157	34167	34171	34177	34185	34192
34197	34240	34253	34259	34263	34267	34273	34285	34294	34297	34301	34309
34314	34318	34349	34362	34393	34405	34440	34458	34462	34498	34501	34504
34547	34560	34654	34658	34666	34676	34679	34682	34685	34692	34701	34704
34710	34736	34740	34744	34754	34758	34872	34876	34964	34974	35030	35038
35050	35054	35062	35118	35176	35247	35265	35268	35287	35293	35314	35328
35334	35342	35353	35372	35387	35398	35402	35411	35417	35433	35543	35566
35570	35574	35644	35669	35770	35778	35782	35797	35801	35809	35813	35892
35896	35948	35951	35954	35961	35967	35971	35986	35991	35996	35999	36010
36015	36026	36034	36043	36046	36051	36060	36063	36068	36072	36076	36081
36087	36096	36101	36106	36111	36115	36120	36131	36139	36146	36151	36156
36183	36248	36267	36287	36293	36299	36307	36335	36345	36350	36353	36618
36625	36732	36791	36796	36895	36908	36911	36919	36965	37104	37143	37181
37187	37194	37199	37204	37230	37235	37240	37246	37251	37265	37270	37274
37278	37282	37287	37311	37316	37321	37334	37340	37384	37426	37431	37486
37512	37536	37559	37585	37628	37634	37640	37646	37671	37690	37708	37722
37726	37787	37804	37827	37841	37867	37984	38033	38064	38084	38098	38146
38149	38173	38205	38219	38222	38270	38275	38345	38351	38370	38379	38384
38427	38528	38533	38539	38546	38551	38609	38614	38620	38626	38631	38637
38643	38733	38739	38745	38751	38766	38816	38850	38855	38897	38905	38927
38961	39024	39033	39038	39043	39074	39120	39157	39162	39222	39381	39385
39441	39445	39449	39452	40086	40090	40149	40153	40168	40172	40190	40194
40212	40221	40225	40237	40406	40431	40435	40497	40502	40527	40541	40547
40551	40560	40571	40589	40597	40605	40625	40646	40650	40857	41251	41276
41301	41305	41362	41384	41404	41408	41480	41519	41647	41651	41668	41672
41708	41712	41718	41722	41952	41973	41979	41990	41995	42002	42007	42041
42044	42076	42090	42107	42118	42124	42133	42171	42426	42489	42507	42643
42670	43032	43037	43041	43046	43051	43057	43083	43087	43097	43139	43235

	44349	44353	44363	44373	44378	44383	44387	44392	44397	44402	44409	44431
	44437	44820	44825	44841	44046	44851	44856	44860	44865	44870	44877	44897
	44933	44945	44950	44954	44959	44964	44969	44973	44978	44985	44997	45062
	45079	45083	45088	45093	45097	45102	45107	45114	45168	45173	45193	45200
	45206	45211	45216	45221	45226	45233	45275	45280	45309	45314	45319	45324
	45331	45355	45360	45369	45379	45384	45389	45393	45398	45403	45408	45415
	45442	45447	45456	45466	45471	45476	45480	45485	45490	45495	45502	45510
	45516	45540	45589	45594	45603	45612	45617	45622	45626	45631	45636	45641
	45648	45658	45673	45677	45702	45707	45722	45727	45732	45736	45741	45746
	45751	45758	45766	45822	45838	45843	45848	45853	45857	45862	45867	45874
	45898	45904	45926	45932	45937	45943	45948	45954	45960	45967	46008	46013
	46029	46034	46039	46043	46048	46053	46058	46065	46077	46151	46155	46201
	46212	46218	46230	46235	46276	46279	46286	46364	46478	46495	46521	46560
	46720	46807	46810	46818	46822	46827	46838	46874	46980	46983	47066	47082
	47135	47278	47281	47326	47336	47341	47369	47393	47405	47411	47422	47493
	47498	47503	47508	47559	47570	47629	47634	47639	47646	47651	47654	47666
	47670	47679	47684	47689	47736	47740	47744	47749	47776	47780	47797	47814
	47844	47848	47851	47856	47859	47862	47865	47886	47889	47892	47935	47942
	47947	47963	47969	47978	47982	47991	48036	48043	48048	48084	48087	48090
	48093	48096	48099									
NOP	3289 #											
RLONG	3291 #	5352	5362	5370	5375	5413	5417	5431	5441	5451	5546	5552
	5655	5760	5776	5960	6138	6145	6152	6163	6177	6251	6694	6748
	6756	6772	6779	6787	6795	6800	6814	6817	6828	6843	6867	6871
	6875	6889	6895	6903	6910	6917	6922	6925	6935	6959	7017	7051
	7113	7116	7119	7122	7125	7129	7198	7207	7271	7290	7294	7638
	7686	7733	7737	7781	7786	7899	8100	8121	8146	8150	8163	8403
	8415	8455	8467	8496	8510	8526	8535	8552	8726	8897	8903	8981
	8989	9031	9034	9043	9048	9058	9066	9071	9098	9103	9111	9115
	9121	9269	9274	9279	9285	9290	9295	9376	9395	9399	9485	9494
	9514	9518	9522	9526	9665	9686	9720	9725	9730	9741	9747	9754
	9760	9763	9773	9781	9834	9839	10692	10697	10702	10751	10763	10770
	10773	10820	10829	10872	11101	11106	11110	11114	11238	11250	11258	11319
	11445	11454	11458	11550	11561	11633	11639	11650	11760	11774	11787	11803
	11816	11896	11907	11913	11928	11933	11982	12051	12056	12127	12140	12227
	12336	12347	12352	12358	12378	12389	12395	12404	12410	12541	12565	12569
	12603	12606	12611	12645	12693	12838	12881	12949	12962	12982	13056	13073
	13151	13165	13227	13244	13312	13317	13323	13342	13346	13355	13360	13366
	13374	13448	13460	13466	13472	13590	13602	13615	13635	13640	13663	13688
	13706	13761	13771	13780	13796	13861	13866	13949	13954	13959	13968	13973
	13977	13986	13990	14038	14043	14052	14062	14073	14212	14258	14304	14314
	14323	14339	14418	14434	14508	14519	14645	14649	14655	14683	14695	14716
	14720	14739	14744	14748	14757	14777	14802	14807	14814	14819	14850	14857
	14866	14872	14935	14959	14964	15118	15140	15151	15156	15161	15175	15182
	15197	15215	15225	15231	15242	15308	15318	15356	15476	15486	15493	15678
	15693	15698	15706	15729	15746	15752	15786	15868	15892	15909	15985	16005
	16010	16013	16025	16039	16047	16052	16124	16128	16246	16273	16277	16419
	16452	16460	16466	16477	16496	16501	16520	16528	16538	16543	16555	16578
	16628	16633	16649	16654	16780	16865	16874	16892	16910	16915	16922	16937
	16948	16954	16973	16978	17004	17025	17048	17070	17100	17110	17155	17164
	17232	17253	17274	17278	17294	17299	17311	17316	17321	17326	17332	17441
	17448	17454	17460	17466	17470	17477	17481	17499	17507	17511	17516	17527
	17534	17539	17546	17581	17587	17595	17613	17625	17638	18221	18240	18244
	18278	18290	18395	18423	18427	18430	18435	18439	18442	18447	18451	18454
	18659	18663	18692	18698	18702	18707	18714	18726	18729	18738	18742	18750

Cross Reference Listing - Field Names and Defined Values

18754	18765	18769	18772	18970	18975	18980	18985	18990	18995	19000	19004
19047	19060	19070	19199	19203	19207	19211	19215	19219	19335	19394	19448
19455	19513	19524	19533	19539	19545	19553	19607	19611	19615	19619	19623
19627	19631	19635	19639	19658	19663	19668	19673	19678	19683	19688	19692
19701	19715	19722	19738	20103	20144	20160	20164	20168	20183	20190	20203
20219	20233	20414	20418	20444	20462	20471	20475	20503	20508	20513	20518
20556	20560	20564	20579	20584	20589	20597	20730	20734	20738	20745	20764
20767	20773	21010	21015	21025	21035	21040	21106	21122	21126	21131	21152
21158	21221	21228	21239	21243	21248	21267	21272	21275	21361	21442	21449
21458	21512	21517	21531	21535	21539	21549	21584	21597	21650	21673	21686
21762	21775	21780	21782	21804	21813	21852	21878	21881	21899	21972	21983
22008	22024	22050	22098	22418	22424	22435	22446	22457	22467	22473	22477
22508	22514	22525	22567	22574	22581	22603	22609	22615	22623	22627	22630
22652	22673	22683	22687	22697	22704	22715	22721	22725	22745	22748	22751
22772	22801	22804	22929	22933	22946	22952	22955	22985	22988	23007	23017
23021	23056	23090	23110	23115	23119	23141	23144	23154	23158	23190	23196
23201	23210	23214	23230	23234	23261	23265	23269	23292	23295	23319	23324
23329	23338	23342	23353	23464	23493	23497	23500	23509	23514	23526	23530
23535	23540	23544	23550	23669	23674	23693	23699	23730	23734	23782	23786
23821	23825	23833	23838	23844	23848	23875	23880	23884	23888	23897	23903
23943	23974	23997	24004	24010	24019	24024	24078	24088	24092	24098	24107
24112	24189	24215	24219	24238	24245	24323	24327	24333	24433	24442	24446
24450	24456	24489	24500	24506	24515	24536	24551	24557	24560	24565	24569
24572	24575	24629	24632	24644	24650	24672	24678	24688	24697	24715	24718
24721	24792	24815	24859	24863	24876	24883	24956	24963	24979	24983	25001
25084	25093	25161	25168	25177	25182	25215	25249	25255	25262	25312	25325
25394	25601	25620	25625	25629	25725	25800	25817	25821	25827	25832	25837
25842	25931	25935	25953	25957	26088	26095	26099	26123	26125	26135	26153
26168	26176	26184	26216	26345	26353	26357	26417	26423	26435	26439	26444
26472	26475	26480	26483	26486	26489	26496	26512	26603	26607	26615	26620
26623	26626	26819	26827	26835	26838	26841	26871	26880	27007	27029	27032
27035	27038	27071	27075	27086	27097	27136	27140	27177	27182	27186	27258
27264	27271	27275	27292	27298	27303	27315	27325	27328	27417	27420	27431
27435	27494	27498	27506	27520	27540	27668	27671	27674	27682	27692	27696
27711	27720	27737	27752	27787	27791	27922	27941	27945	27959	27970	27979
27992	28003	28017	28021	28030	28060	28069	28078	28166	28217	28397	28401
28411	28558	28573	28686	28691	28713	28721	28733	28852	28858	28942	28945
28956	28962	28971	28978	28982	29086	29097	29166	29208	29218	29360	29402
29409	29457	29545	29549	29578	29615	29636	29649	29654	29734	29739	29742
29745	29754	29758	29808	29831	29854	29927	29930	29934	29953	29968	29973
29977	29981	29986	29991	29996	30001	30005	30009	30011	30019	30042	30049
30054	30083	30089	30094	30099	30108	30113	30174	30180	30185	30189	30205
30220	30324	30331	30336	30342	30384	30391	30395	30400	30427	30433	30438
30443	30484	30488	30521	30595	30644	30655	30658	30664	30668	30673	30723
30728	30896	30902	30936	30941	30946	30982	30990	31032	31119	31123	31126
31133	31139	31163	31166	31175	31178	31181	31333	31411	31421	31426	31430
31435	31469	31479	31617	31622	31626	31629	31634	31740	31748	31779	31853
31857	31860	31874	31884	31891	31901	31911	31914	31920	31923	31929	31932
31938	31941	31944	31947	32093	32151	32158	32176	32200	32203	32206	32210
32237	32240	32243	32247	32445	32466	32485	32501	32514	32525	32719	32723
32727	32730	32733	32737	32794	32888	32893	32906	33115	33131	33134	33137
33146	33164	33167	33171	33175	33473	33500	33503	33514	33518	33522	33525
33549	33553	33556	33571	33623	33805	33835	33893	33948	33959	34008	34161
34279	34290	34304	34354	34358	34368	34372	34387	34408	34446	34450	34509
34542	34544	34550	34553	34565	34594	34599	34604	34607	34613	34619	34623

```
                    34661   34669   34696   34724   34728   34732   34952   34960   34977   34990   34999   35008
                    35017   35094   35179   35221   35356   35579   35621   35650   35659   35746   35774   35790
                    35908   35957   36012   36125   36128   36142   36227   36232   36237   36241   36258   36263
                    36271   36275   36279   36283   36295   36311   36327   36331   36341   36357   36465   36469
                    36474   36491   36549   36553   36558   36575   36622   36705   36718   36729   36736   36739
                    36759   36763   36767   36771   36793   36802   36817   36878   36891   36914   36922   37226
                    37294   37298   37302   37306   37326   37330   37356   37364   37436   37440   37479   37651
                    37678   37747   37862   37872   37908   37911   37917   37980   37989   38163   38262   38292
                    38296   38301   38354   38357   38362   38393   38437   38449   38492   38566   38658   38761
                    38775   38922   38970   39074   39011   39028   39066   39070   39078   39082   39090   39117
                    39152   39166   39184   39335   39338   39341   39403   39407   39411   39415   39419   39462
                    39474   39493   39497   39500   39507   39692   39722   39736   39801   39809   39874   39908
                    39918   39943   39954   39964   39974   39996   40033   40039   40054   40074   40079   40140
                    40157   40162   40244   40313   40336   40375   40380   40398   40420   40425   40566   40584
                    40609   40639   40642   40692   40701   40705   40713   40718   40755   40762   40766   40774
                    40778   40814   40821   40862   40873   40901   40929   41412   41418   41433   41438   41457
                    41603   41746   41755   41766   41835   41926   41947   42081   42102   42252   42285   42318
                    42379   42402   42541   42557   42579   42603   42652   42658   42715   43103   44367   44424
                    44454   44483   44527   44544   44578   44623   44640   44668   44717   44729   44757   44835
                    44889   44913   45010   45031   45038   45066   45073   45136   45147   45185   45245   45290
                    45373   45451   45460   45523   45533   45564   45598   45607   45669   45717   45772   45780
                    45800   45832   45908   45919   46023   46187   46215   46226   46295   46348   46357   46360
                    46394   46410   46419   46435   46444   46460   46466   46528   46533   46538   46550   46555
                    46771   46775   46869   46915   46996   47613   47619   47695   47762   47766   47772   47788
                    47793   47828   47832   47835   47839   47873   47877   47881   47908   48000   48027
RSIZE               3290 #  6590    8306    8350    8356    8367    8377    8435    8440    8521    8589    8595
                    8601    8619    8635    8678    8684    8690    8708    8762    8768    8775    8806    8812
                    8819    8850    8856    8864    8913    8930    8941    8985    9318    9327    9335    9339
                    9505    11478   11485   11667   11675   11686   11696   11703   11807   13673   14782   14952
                    15799   15850   15916   16029   16073   16211   16267   16510   16548   16567   16593   16637
                    16740   16753   16767   16808   16823   16994   17030   17052   17242   17303   17491   19341
                    24640   24948   24969   24997   25102   25171   25401   28959   29092   31169   31864   33244
                    33782   34011   35359   35982   36018   36056   36160   38827   38832   38837   38844   39002
                    39006   39053   39060   39096   39103   40989   40999   41005   41024   41035   41042   41062
                    41073   41080   41100   41111   41118   41174   41184   41231   41257   41268   41279   41314
                    41321   41327   41332   41447   41451   41485   41493   41524   41532   41566   41571   41599
                    41607   41626   41655   41761   41770   41819   41829   41858   41890   42014   42019   42026
                    42032   42037   42147   42158   42163   42209   42220   42247   42313   42388   42447   42477
                    42619   42629   42776   42782   42838   42845   42864   42925   42931   42984   42991   43005
                    43066   43072   43125   43130   43157   43166   45547   47396
WCTRL               3295 #
    ACLOSYNC        3299 #
    ASTLVL          3305 #  35782
    ASTLVL_WB       3306 #  5933    35256   36729
    CLRCH.VA_WB     3358 #  5997    9170    9175    9180    9185    9191    9195    9199    9203    9209    9214
                    9219    9224    9230    9234    9238    9242    9554    9565    9581    9591    9596    9600
                    9608    9614    9619    9625    9630    9634    9642    9648    9653    12852   13081   14209
                    14220   14235   14255   14673   15783   35520
    CLRTB.VA_WB     3356 #  9170    9175    9180    9185    9191    9195    9199    9203    9209    9214    9219
                    9224    9230    9234    9238    9242    9554    9565    9581    9591    9596    9600    9608
                    9614    9619    9625    9630    9634    9642    9648    9653    12852   13081   14209   14220
                    14235   14255   14673   15783   35696
    CM.TP.FPD.F5.STEPC  3318 #  20800   21314   29846
    CM.TP.FPD.FLAGS 3320 #  21983   22024   22050   23743   23747   23751   23791   23796   23857   23861   23865
                    24660   25171   25725   26871   26880   28959   29208   29765   31169   31309   31740   31748
```

Cross Reference Listing - Field Names and Defined Values

	31864	33244	39033									
CONREAD	3333 #	6135	7954	8018	8040	8045	36087	38205	38219			
CONWRITE	3332 #	5903	5924	5966	6243	6247	6707	6990	7967	20975	35445	38225
FLAGS_WB	3319 #	12881	13073	13866	24778	25015	25207	25371	28263	28268	29201	29376
	31224	32029	34033	34707								
FPA.ENABLE_WB5	3300 #	35551										
FPTCR	3304 #	37208	38533	38893								
GRANT	3314 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
	9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
	9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
	14235	14255	14673	15783	38146	38146						
INIR	3324 #											
INIR_WB	3326 #	35342										
IPL_WB	3310 #	35026	37374	37400	37563							
IPR	3311 #											
LOADCRAR	3331 #	5898	5912	5963	6132	6232	6977	7943	7958	8013	8024	20972
	35082	35090	35459	35833	35837	35841	38202	38217	47430			
LOADTRAR	3335 #	5488	5951	5957	35066	35078	35381	35385	35477	35817	35821	35825
	38095	38156										
MBUS_WDR	3354 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
	9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
	9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
	14235	14255	14673	15783	39722	39722	39736	39736				
MDR_0	3350 #	9148	9153	9158	9163	9170	9175	9180	9185	9191	9195	9199
	9203	9209	9214	9219	9224	9230	9234	9238	9242	9554	9565	9581
	9591	9596	9600	9608	9614	9619	9625	9630	9634	9642	9648	9653
	9669	10770	10873	10996	11106	11233	11238	11243	11352	12852	12983	13063
	13081	13318	13711	13954	13973	13991	14209	14220	14235	14242	14255	14634
	14673	14695	14701	14739	14850	14857	15072	15078	15088	15094	15103	15118
	15205	15783	16995	17243	17528	18227	18363	18371	18671	13678	30658	35750
	35755	35758	41594	41623	45569	45678	46187	46348	46944	47062	47147	47385
	47389	47978										
MDR_IR	3351 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
	9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
	9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
	14235	14255	14673	15783	32041	46903						
MDR_WB	3349 #	6764	6782	6790	6806	6839	6883	8153	9170	9172	9180	9185
	9191	9195	9199	9203	9209	9214	9219	9224	9230	9234	9238	9242
	9554	9565	9581	9591	9596	9600	9608	9614	9619	9625	9630	9634
	9642	9648	9653	9791	10679	10732	10804	10816	10825	11166	11177	11383
	11386	11470	11652	11662	11756	12033	12040	12744	12852	13081	13571	13576
	13581	13595	13629	13645	13651	13657	13677	13692	13715	14209	14220	14235
	14255	14673	14947	15108	15113	15187	15191	15220	15736	15783	16069	16234
	16242	16501	19009	19698	20207	20211	20243	20252	20261	20269	20278	20287
	20296	20304	20313	20322	20331	20339	21164	22496	22556	25030	25095	30252
	30276	30280	30346	30468	30509	31248	31252	31256	31260	32442	32463	32498
	34610	34663	36173	38373	38456	38460	39889	39895	39899	39912	39935	39938
	39979	39983	39987	39990	40241	40265	40319	40333	40444	40905	40909	40913
	40933	40937	40941	41021	41059	41097	41135	41206	41221	41225	41247	41587
	41678	41683	41742	42191	42195	42199	42735	42814	42859	42883	44348	44430
	44472	44476	44483	44492	44496	44500	44550	44571	44578	44588	44657	44661
	44668	44678	44682	44686	44746	44757	44766	44770	44774	44798	44819	44932
	44963	44991	44996	45010	45021	45061	45073	45092	45120	45125	45354	45424
	45515	45657	45682	45977	45987	47958	47973	47988	48006	48012	48017	48021
MEMSCAR_WB	3360 #	5519	6142	6235	6685	6972	7013	35106	35110	35122	35497	35502

Cross Reference Listing - Field Names and Defined Values

	35511	35525	35849	35852	35856	35860	35872	35918	35922	35926	35930	35934
	35938	37109	37661	37801	37839	37853	37914	37961	37964	37968	37972	37976
	39973	39930	40303	40509	46148	47157	47168	47181	47187	47200	47214	47227
	47240	47254	47265	47271	47401	47415						
MEMSCR	3361 #	6145	35543	35948	36096	36120	37690	37708	37804	37827	37841	37917
	38033	38149	38362	38449	39943	39954	47278					
MEMSCR_WB	3362 #	5523	5605	6277	6651	6769	6987	7021	7160	7389	7398	7416
	7426	35450	35507	35546	35579	37733	37807	37813	37819	37856	38160	40306
	40316	40512	40521	46159	47184	47190	47230	47268	47274	47284	47419	
NOP	3298 #											
PC_PC+WB	3342 #	15340	18914	19096	19187	19248	19321	19409	19413	19417	19460	19464
	19473	19492	19564	19582	42356	42492						
PC_WB	3341 #	5357	5421	5454	5458	5462	5667	5675	5880	6122	6138	6436
	6656	6701	6969	7027	7041	7045	7190	16293	17470	17486	17578	17616
	19021	19719	20458	20594	20913	20962	22457	22627	22642	22673	22721	22788
	22994	23049	23053	23106	23186	23449	23453	23526	23550	23760	23807	23816
	23875	23928	24014	24102	24250	24317	24327	24371	24427	24569	24712	24789
	24843	24859	24873	25122	25376	25388	26189	26198	26306	26333	27723	27727
	28849	28865	29227	29238	29338	29369	30025	31126	31244	31285	31290	31299
	31305	32037	33556	33575	33956	34556	34616	34673	35593	35616	36527	36715
	36841	37255	37344	37379	37388	37755	37776	38105	38806	39209	39268	39434
	39644	39755	40255	40390	40994	41030	41068	41106	41179	41263	41489	41528
	41612	41726	41781	41805	41824	41867	41899	42215	42257	42322	42397	42434
	42438	42442	42464	42624	42655	42854	43000	43079	43190	44750	45001	46243
	46251	46305	46313	46391	46407	46416	46432	46441	46457	46475	46492	46569
	46576	46583	46590	46595	46601	46608	46615	46622	46629	46635	46642	46648
	46655	46662	46668	46675	46682	46689	46696	46744	46886	46893	46895	46911
	46922	46925	46947	46993	47003	47012	47022	47066	47094	47101	47112	47153
	47165	47176	47197	47211	47222	47237	47251	47262	47293	47302	47356	47378
PME	3348 #	5939	35143	36755								
PREV_CUR.ISCUR_WB	3308 #	36348	36635	36639	38282	40869						
PREV_WB	3309 #											
READCRAR	3334 #											
READTRAR	3338 #											
REICHK	3313 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
	9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
	9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
	14235	14255	14673	15783	39364	39365						
REVLEVEL	3301 #	35896										
SOFTIPR_WB	3307 #	5927	35275	35308	37491	39397						
STEPC_WB	3317 #	7173	7949	8008	14112	14500	14505	14665	14970	15775	21262	25608
	25655	25858	25863	26171	26180	26192	26317	26321	26578	26588	26641	27356
	27360	27700	29349	29488	29828	30574	30589	30640	30840	30883	30892	31048
	31517	31522	32063	32067	35741	38952	46352	46763				
TB_WB	3355 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
	9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
	9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
	14235	14255	14673	15783	35584	35598	35612	35624	35701	36678	37810	37816
	40249	40365	40441	40655	47193	47204	47233	47244				
TCSR.IICR	3325 #	8035	35050	35054	35801	35809	38816	38897	38905			
TCSR_WB	3323 #	5929	35337	35368	35375	38851						
TODCLK	3328 #	36010										
TODCLK_WB	3327 #	35062	35411	35417	35427	35430	35436	35999	36004	36046		
TU58READ	3337 #	5491	35387	36081	38084	38098						
TU58WRITE	3336 #	5495	5507	5954	5960	35390	35393	35440	35492	38109		

Cross Reference Listing - Field Names and Defined Values

UVCTR_CM.IS	3312 #	34883	34952	35746	36268	36674	36772	36848	36892	37236	37241	37266
	37288	38259	39381	39437								
VA_PC+1+W.PC_PC+1	3345 #	42797	42947	43087	43202	44372	44391	44488	44504	44583	44600	44673
	44690	44762	44778	44840	44859	44949	44968	45078	45096	45192	45215	45295
	45313	45378	45397	45465	45484	45611	45630	45721	45740	45837	45856	45925
	45947	46028	46047									
VA_VA+4	3344 #	5634	5680	5732	5790	5800	8411	8532	9055	9063	9760	9769
	9787	9804	11922	14798	15550	15719	15758	16020	16035	16218	16610	16617
	16641	16776	16888	17009	17013	17018	17039	17081	17089	17260	17265	17284
	18409	18413	18417	18702	18726	20543	20581	20586	20591	20612	20621	20631
	20641	20651	20661	20670	20680	20690	20700	20710	20719	20761	21461	21527
	21708	28159	29397	30930	36705	36711	36719	36735	36758	36762	36865	36926
	36932	36938	39649	39692	39821	39828	40038	40380	45509	45522	45541	45653
	45765	45774	45973	46070	46078	46767	46875	47056				
VA_WB	3343 #	5526	5546	5552	5611	5659	5698	5712	5726	5749	5760	5796
	5887	5907	5917	6746	6871	6875	7183	7219	8127	8131	8135	8139
	8455	8467	9498	9510	9778	12579	12641	14762	14789	14794	15714	15991
	16140	16284	16532	16926	16942	17075	17159	17168	17172	17176	17550	18405
	18692	18718	19064	19148	19528	19639	19722	19733	20098	20123	20134	20190
	20195	20203	20444	20503	20508	20513	20530	20536	20569	20734	20738	20745
	20817	20819	20899	20929	21102	21111	21152	21158	21206	21216	21232	21253
	21272	21434	21438	21442	21449	21453	21467	21471	21512	21521	21531	21535
	21542	21546	21584	21588	21600	21607	21615	21621	21650	21655	21662	21668
	21678	21681	21693	21701	21723	21730	21758	21762	21768	21790	21794	21801
	21808	21848	21852	21858	21870	21874	21881	21892	21896	21903	21916	21988
	22046	22446	22772	22991	23060	23064	23115	23119	23196	23201	23248	23265
	23269	23324	23329	23540	23544	23555	23562	23825	23833	23838	23884	23888
	24195	24572	24808	24819	24831	24854	25113	25128	25322	25363	25660	25691
	25717	25806	25868	25872	26337	26341	26593	26645	26670	26674	26678	26682
	26876	26899	27048	27341	27704	28109	28121	28154	28208	28333	28337	28660
	28665	29162	29213	29275	29355	29365	29578	29615	29643	29679	29684	29697
	29721	29726	29831	29842	29911	30039	30118	30215	30359	30363	30447	30630
	30635	30713	30755	30759	30873	30878	30887	30908	30926	30967	31133	31336
	31348	31527	31719	31752	32375	32384	32395	32399	32403	32482	32511	32813
	32822	32828	32832	32836	33018	33496	33506	33945	34368	34432	34446	34450
	34489	34509	35126	35130	35135	35588	35629	35884	36136	36293	36303	36315
	36341	36357	36486	36570	36643	36667	36683	36700	36745	36781	36785	36808
	36817	36853	36861	36871	36878	36882	36888	37114	37119	37129	37135	37330
	37356	37364	37482	37678	37685	37763	37768	37773	37876	37980	37989	38091
	38112	38167	38213	38228	38278	38492	38561	38566	38577	38653	38658	38721
	38770	38775	38916	39086	39166	39179	39184	39189	39329	39335	39725	39739
	39759	39805	39813	39833	39884	39948	39959	40049	40059	40065	40070	40095
	40184	40207	40260	40300	40309	40321	40327	40330	40368	40393	40439	40506
	40515	40531	40537	40576	40593	40630	40636	40713	40722	40774	40781	42402
	42557	42579	42597	42611	42706	42719	42743	42748	42752	42757	42762	42767
	42776	42782	42788	42792	42864	42891	42896	42900	42905	42910	42916	42925
	42931	42938	42942	42958	43005	43032	43037	43041	43046	43051	43057	43072
	43083	43097	43139	43147	43197	43207	43225	43230	43235	44352	44367	44377
	44382	44386	44396	44401	44448	44507	44511	44522	44533	44567	44592	44596
	44603	44607	44618	44629	44693	44697	44708	44713	44781	44785	44804	44824
	44835	44845	44850	44855	44864	44869	44908	44972	44977	45101	45106	45172
	45185	45199	45205	45210	45220	45225	45251	45279	45290	45300	45305	45308
	45318	45323	45359	45373	45383	45388	45392	45402	45407	45446	45460	45470
	45475	45479	45489	45494	45558	45593	45607	45616	45621	45625	45635	45640
	45706	45717	45726	45731	45735	45745	45750	45785	45794	45821	45832	45842

Cross Reference Listing - Field Names and Defined Values

		45847	45852	45861	45866	45903	45919	45931	45936	45942	45953	45959	46012
		46023	46033	46038	46042	46052	46057	46165	46179	46814	46831	46882	46889
		46907	46918	46951	46989	46999	47008	47015	47018	47090	47097	47108	47139
		47161	47172	47207	47218	47247	47258	47298	47326	47336	47341	47369	47808
WDR_WB		3352 #	5563	5568	5578	5595	5630	5671	6900	6907	6914	7226	8291
		8350	8357	8362	8373	8403	8415	8420	8435	8441	8459	8471	8501
		8521	8526	8535	8557	8596	8606	8620	8652	8685	8695	8709	8731
		8769	8780	8813	8824	8857	8869	8914	8931	8942	8986	9044	9049
		9072	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
		9224	9230	9234	9238	9242	9319	9328	9336	9340	9501	9514	9518
		9522	9526	9554	9565	9581	9591	9596	9600	9608	9614	9619	9625
		9630	9634	9642	9648	9653	9687	9721	9725	9730	9736	9742	9749
		9754	9764	9774	9782	9834	9839	10693	10698	10703	10753	10765	10775
		10821	10830	11259	11479	11486	11668	11676	11687	11697	11704	11776	11789
		11897	11908	11914	11929	11934	12852	13081	13167	13246	13772	13797	14209
		14220	14235	14255	14315	14340	14673	14721	14749	14767	14772	14803	14808
		14815	14820	15319	15487	15494	15783	16741	16754	16769	16781	16810	16825
		16866	16875	16893	16931	16949	16955	18241	18245	18278	18291	19291	19297
		19389	19449	19643	19726	20197	20229	20239	20248	20257	20265	20274	20283
		20292	20300	20309	20318	20327	20335	20344	20424	20433	20441	20453	20468
		21036	21041	21147	21361	21457	21464	21482	21538	21550	21611	21618	21624
		21705	21711	21726	21733	21805	21812	21900	21906	21919	21997	22002	22015
		22057	22092	23045	23102	23182	23256	23306	23310	23473	23520	25713	26211
		26895	28171	28296	28593	28617	28622	28632	28809	28823	28827	28831	28835
		29692	29702	29706	30000	30045	30125	30130	30265	30271	30284	30289	30355
		30373	30377	30387	30531	30535	30609	30618	30708	30745	30852	30856	30860
		30864	30923	30957	30962	30972	30977	31744	32843	32847	32925	32929	32964
		32969	33011	33030	33040	33952	34435	34494	36360	36776	36812	36857	36875
		36885	36899	36911	36929	36935	36941	37352	37360	37368	37993	37997	38001
		38005	38009	38039	38482	38486	38570	38662	40989	40999	41005	41025	41036
		41043	41063	41074	41081	41101	41112	41119	41191	41213	41258	41269	41280
		41499	41538	41566	41571	41627	41656	41762	41771	41820	41830	41836	41852
		41859	41884	41891	42210	42221	42241	42307	42375	42383	42393	42537	42545
		42550	42565	42570	42620	42630	46170	46175	46870	46878	47052	47059	47144
		47331	47359	47381									
WDR_WB.UR		3353 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
		9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
		9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
		14235	14255	14673	15783	18711	18722	18733	19108	19112	19116	19120	19124
		19128	39729	39743	39816	40179	40217	40232					

; This page intentionally left blank.

Cross Reference Listing - Macro Names

```
CLEAR MOPZERO(FLAG1)        3706 #  15679  15753  15759  15765  13829
CLEAR MUL1(FLAG2)           3707 #  12511  12517  12968  13479  14518  14677  14851  14985  15824  15887
CLEAR MUL2(FLAG3)           3708 #  12525  12532  12989  14037  14423  14688  14858  14990  15006  15013  15787
                            15898  15904
CLEAR OPZERO(FLAG3)         3709 #  12161  12228  13361
CLEAR OVER(FLAG2)           3710 #  15166
CLEAR POP1C(FLAG4)          3711 #  48033
CLEAR READ(FLAG1)           3712 #  16034  16473  16569  16582  16594  16642  17019  17080  17244  17304
CLEAR REGINT(FLAG1)         3713 #  14867
CLEAR SAMESIGN(FLAG4)       3714 #  12359  12704  12710  13664
CLEAR STACK FLAG            3715 #  6239   37227  37361  37369  38010  38040  38297  38302  38663  38917  39652
                            39947  39958
CLEAR SUB(FLAG1)            3716 #  13158  13234
CLEAR TP                    3717 #  37188  38292  38649  38932  39173
CLEAR WRITE(FLAG1)          3718 #  11251  13473
CLOBBER MTEMP0              3719 #  45062  45093  45107  45173  45211  45226  45480  45594  45626  45641  45658
                            45707  45736  45751  45943  46043
CLOBBER MTEMP0 DEF          3720 #  44349  44353  44363  44373  44378  44383  44387  44392  44397  44402  44431
                            44437  44820  44825  44841  44846  44851  44856  44860  44865  44870  44897  44933
                            44945  44950  44964  44969  44973  44978  44997  45079  45097  45102  45193  45200
                            45206  45216  45221  45275  45280  45309  45314  45319  45324  45355  45360  45369
                            45379  45384  45389  45393  45398  45403  45408  45447  45456  45466  45471  45476
                            45485  45490  45495  45510  45516  45540  45612  45617  45622  45631  45636  45722
                            45727  45732  45741  45746  45766  45822  45838  45843  45848  45853  45857  45862
                            45867  45904  45926  45932  45937  45948  45954  45960  46013  46029  46034  46039
                            46048  46053  46058  46077
CMP SIGNS?                  5119 #  8910   9039   9324   9416   9425   41953  42108
COMPLETE CPU BUS CYCLES     3786 #  5886   5911   5918   6005   35131  36964  37798  3888   38901  40094  40183
                            40206
CONREGS_D_M[]_R[]           3882 #  5903
CONREGS_M[]                 3883 #  6990
CONREGS_M[].OR.ZLIT16[]     3884 #  38225
CONREGS_M[].RR.16           3885 #  7967   35445
CONREGS_M[]_R[]             3886 #
CONREGS_R[]                 3887 #  5924
CONREGS_ZLIT16[]            3888 #  5966   6243   6247   6707   20975
COUNT OR INT TIMER?         5120 #  15918  16213  17033  17055  17305  17494  25649  25791  26582  28418  32081
                            32123
CRAR_ZLIT16[]               3890 #  5898   5912   5963   6132   6232   6977   7943   7958   8013   8024   20972
                            35082  35090  35459  35833  35837  35841  38202  38217  47430
D(OD)_ZLIT0[]               3892 #
DBZ STEPC?                  5123 #  7042   7982   9191   9230   9591   9625   12851  13080  14208  14219  14234
                            14254  14462  14480  14672  15782  17540  21268  26330  28485  29508  29515  29843
                            29912  30600  30610  30614  30619  30631  30714  30786  35437  35697  36706  36866
                            38971  38975  46254  46316  46411  46436  46461  46495  46768  47386  47983
DEC STEPC                   3722 #  5904   5910   6148   6814   6886   6889   6973   7021   7027   9600   9608
                            9634   9642   20146  20170  20565  20779  20787  20805  21319  28410  28416  28427
                            28746  28749  30665  30669  30674  30724  30903  30909  30924  35402  35405  35649
                            35658  36000  46151  47397  47411
DIVDA SOR IN R[]            3723 #  14478
DIVDS SOR IN R[]            3724 #  14460
DIVFAST+ SOR IN R[]         3725 #  9552   9589   9594   9599   9606   9612   9617   13079
DIVFAST- SOR IN R[]         3726 #  9563   9579   9623   9628   9633   9640   9646   9651
DIVIDEND SIGN?              5122 #  42149
DSIZE?                      5124 #  9149   9154   9159   9164
```

```
D_(M[] R[]).RR.9                3893 #
D_(M[] R[]).RR.P                3894 #
D_(M[]+CONX(2)).SR.1            3895 #
D_(M[].RR.P).AND.R[]            3896 #  29859   29885
D_(R[] M[]).RL.P                3897 #  9735
D_(R[]+CONX(2)).SR.1            3898 #
D_-1                            3899 #
D_CONX.SIZ                      3900 #  42772   42920   43061   43116
D_D+OLIT24[]                    3901 #  5614
D_D+R[]                         3902 #  9489    42153
D_D+R[]+ALKC                    3903 #
D_D+ZLITO[]                     3904 #  5630    12341
D_D-1                           3905 #  5947    27823
D_D-CONX(2)                     3906 #  27964
D_D-CONX(4)                     3907 #  27995
D_D-R[]                         3908 #  23907
D_D-ZLITO[]                     3909 #  6006    28027   28075   30987   30997   31001   31007   31019   31025   35531   35535
D_D.AND.OLIT24[]                3910 #  5625
D_D.AND.ZLITO[]                 3911 #  33819   33831
D_D.AND.ZLIT28[]                3912 #  35331
D_D.OR.R[]                      3913 #  26428
D_D.XOR.Q                       3914 #  46301
D_D.XOR.ZLIT12[]                3915 #  12371   13670
D_M[]                           3916 #  7853    14450   14705
D_M[] SQR                       3917 #
D_M[]+R[]                       3918 #  21020   21864
D_M[]+ZLITO[]                   3919 #  37098   38389   38404   38433   38443   38472   38698   38705   38711   38716   38728
                                38756
D_M[]-R[]                       3920 #  9481    28394   30829   42143
D_M[]-ZLITO[]                   3921 #
D_M[].AND.R[]                   3922 #  30495   30731   30771   39372
D_M[].AND.ZLIT16[]              3923 #
D_M[].OR.(R[].RL.24)            3924 #  36065
D_M[].OR.R[]                    3925 #  26104   26451   31108   31415
D_M[].RR.16                     3926 #  15351   15366
D_M[].RR.16 Q_R[]               3927 #  15519
D_M[].SR.1                      3928 #  25686   31714
D_Q Q_D                         3929 #  15379   35325
D_Q_-T                          3930 #
D_Q_0                           3931 #
D_Q_D.XOR.R[]                   3932 #  46246
D_Q_M[]                         3933 #  22531   22541
D_Q_M[].XOR.Q                   3934 #  46308
D_RNUM_D+ZLITO[]                3935 #  46386   46402   46427   46452   46487   46780
D_RNUM_D-ZLITO[]                3936 #  46182   46367   46543   46729   46747   47069
D_RNUM_ZLITO[]                  3937 #  46162   46344   46716   46724   46739   47062
D_R[]                           3938 #  22949   23483   23992   24001   24437   33788   33792   33798
D_R[]+CONX.SIZ                  3939 #  42829   42975
D_R[]-1                         3940 #  27747
D_R[]-D-ALKC                    3941 #  9573    9584
D_R[]-M[]                       3942 #  23487
D_R[].OR.Q                      3943 #  5646
D_R[].SR.1                      3944 #  26636   27336
D_R[]_ZEXT(XB) PC_PC+1          3945 #
D_SEXT(XB) PC_PC+T              3947 #  19381
```

Cross Reference Listing - Macro Names

```
D_SEXT(XB) PC_PC+2          3949 #  19283  19329
D_ZEXT(M[])                 3951 #  19571  19578  22596  22661  22710
D_ZLIT0[]                   3952 #  12510  12699  12709  12721  12724  14086  14116  15123  15740  16059  16560
                            38419
D_ZLIT0[]-D                 3953 #  12331
D_ZLIT12[]                  3954 #  11885  12516  12703  12715  12729  12734  16063
D_ZLIT16[]                  3955 #  7048
D_ZLIT24[]                  3956 #
D_ZLIT24[] M[]_HARD.REV     3957 #  35896
D_ZLIT4[]                   3959 #  35515
D_ZLIT8[]                   3960 #  5943   5991
EMODH(FLAG4)?               5126 #
EXPONENT RANGE?             5127 #  15104  15131  15206  47953
FLAG0?                      5129 #  5492   5643   6551   6652   6670   6682   6978   7610   7630   7639   7679
                            7683   7690   9195   9234   9596   9630   18762  20445  22935  22964  22980  22994
                            24508  24984  26306  26333  26349  27911  28489  28528  28750  29903  30240  32536
                            32836  32844  32848  32889  32892  32907  32964  32969  33031  33040  33716  33802
                            34154  34181  34264  34275  34280  34291  37369  38011  38487  38664  39453
FLAG1 (FLAG2.XOR.FLAG3)?    5130 #  27342  29749  30379
FLAG1?                      5131 #  7552   9498   17582  20147  20165  20171  20561  20566  23064  23338  23353
                            23674  23876  25709  25957  29693  29735  29953  29968  29973  30114  30118  30281
                            30374  30492  35703  37322  37375
FLAG2?                      5132 #  5967   7968   8051   9553   9559   9565   9569   9580   15079  15095  15157
                            18424  18431  18483  18487  25173  25686  30277  30680  31301  31549  31575  31578
                            33806  34561  37353  42204
FLAG3?                      5133 #  6474   6674   9489   18227  18699  18747  20221  20235  23474  23521  23557
                            23684  23936  24839  24974  25041  25086  25717  25997  26002  26771  26775  26824
                            26842  26884  28284  28537  29525  31674  31692  31752  33793  33799  37357  37365
                            42154  42368  42604
FLAG4?                      5134 #  47937
FLAG<1-0>?                  5135 #  14494  15520  17233  18402  18406  26671  26675  26679  26683  41452  42045
                            43168  43191  43215  46543  46547
FLAG<2-0>?                  5136 #  25953  26636  27336  28805  31714  42707  47974
FLAGS_D_R[]                 3962 #  24778
FLAGS_M[].AND.ZLIT0[]       3963 #  34033  34707
FLAGS_R[]                   3964 #  25207  29201  29376  31224  32029
FLAGS_ZLIT0[]               3965 #  28263  28268
FLUSH_XB                    3728 #  35616  36527  37755  37776  39644  40390
FORCE 32 BITS OF VA         3730 #  40567  40577  40585  40594  40620  40629  40654
FORCE CACHE PARITY          3731 #  47144  47147
FPA PRESENT?                5137 #  35864
FPA(FLAG0)?                 5138 #  39055
FPA.ENABLE_M[].RR.P         3967 #  35551
FPAWAIT                     3729 #  8981   8989   9110   9120   16739  16752  16766  16779  16807  16822  16850
                            16864  16873  16881  16891  16930  16936  16947  16953  17090  17096  17099  17285
FPA_MB_M[]_R[]              3968 #  17195  17199  17210  17214  17218  17221
FPA_M[]                     3969 #  8972   8976   16727  16746  16786  16816  16838  16858  16905  16968  17018
                            17080  17085  17095  17203  17224  17269  45990
FPA_M[] FPA_WB_R[]-0        3970 #  16722  16733  16761  16797  16833  16844
FPA_M[] MDR_R[]             3971 #  45977
FPA_Q_MDR MTEMP0_R[]        3972 #  44409  44877  45415  45898  45967  46008  46065
FPA_Q_M[]                   3973 #  44435  44442  44895  44902  45368  45539  45553  45982  46076  46084
FPA_Q_M[] MDR_Q             3974 #  44430
FPA_Q_M[] MDR_R[]           3975 #  44348  44819  45354
FPA_Q_M[] VA_R[]            3977 #  44352  44386  44401  44824  44855  44869  45359  45392  45407  45903  45959
```

```
                    46012   46057
FPA_Q_M[].LITNXT    3976 #  44361   45913
FPA_R[].SIZ_M[]     3978 #  45547
FPA_R[]_M[]         3979 #  45780
FPA_WB_R[]-0        3980 #  16791   16802
FPD?                5139 #  37835   37850  38390  38405  38473  38729  38757  38894  38913  38918
FPS1?               5140 #  5358    6240
FPS2?               5141 #  5452
FPS3?               5142 #  6991    8025
FRO.FLTZ?           5143 #  10684   10740  10747  10808  10910  10915  10977  10982  11093  11122  11226
                    11551   11640   11761  11983  12052  12128  12839  12950  13057  13152  13228  13449
                    13461   13762   13781  13862  14305  14324  14419  14656  14936  14960  15309  15314
                    15394   15400   15481  15725  15754  15788  16026  16040  16247  16421  16556  16655
                    47614   47620   47696
GFLOAT(FLAG4)?      5145 #  15390   45602
INIR_M[]_Q          3982 #  35342
INTPEND OR TIMER?   5147 #  14087   14117  14465  14483  15741  16561  17044  17289  17503  22439  23025
                    23162   23314   23457  23738  23852  24051  24139  24209  24346  24467  25664  25810
                    26114   26455   26597  27090  27319  27458  27470  27709  27764  28043  28177  28211
                    28297   28431   28465  28754  29503  29631  30684  30868  30951  31137  31447  31473
                    31607   31638   32415  32540  32741  32818  33566  34397  34454  35575  35602  36319
                    36691   37391   37406  38883  40060  40145  40266  40407  47725
IO RESET            3733 #  5323    5328   5334   5947   35562  35565
IP.TS(SIGND CMP)?   5148 #
IP.TS?              5149 #  14077   14108  14471  14475  16629  24036  24125  24337  24457  26104  26444
                    27075   27292   27304  27441  27450  27938  27964  27995  28030  28166  28746  29258
                    29530   29624   30725  31129  31436  31460  31599  31630  32446  32467  32502  32525
                    32733   32794   33572  34373  34388  36312
IPL_M[].RL.16       3983 #  35026
IPL_[]              3984 #  37374   37400  37563
IR<2-0>?            5152 #  9670    10733  11987  12869  12882  13074  13094  18379  18383  19042  19053
                    19077   19093   19098  19167  19184  19189  20105  20187  20429  20469  22458  22677
                    22912   23221   23450  23454  24202  24206  24327  24333  25254  25303  25313  25630
                    25634   25807   25857  25862  26475  26652  26656  26874  31245  32067  39090  42662
IR<2>?              5150 #  8292    8307   15110  19284  19330  20811  26125  26589  42508
IR<5>?              5151 #  11362   11374  11881  11889  12512  12518  12705  12711  12725  12730  14712
                    14769   14774   14779  14784  16021  16943  17106
IRD1                3734 #  8240    8245   8261   8266   8278   8301   8316   8321   8351   8357   8363
                    8368    8373    8378   8416   8420   8436   8441   8460   8472   8501   8511   8522
                    8536    8553    8564   8590   8596   8606   8620   8636   8653   8679   8685   8695
                    8709    8727    8737   8763   8770   8780   8807   8814   8824   8851   8858   8869
                    8942    8946    8950   8992   8996   9072   9319   9336   9519   9527   9687   9721
                    9731    9765    9782   9835   10694  10699  10704  10822  10831  10976  10930  10934
                    11016   11026   11029  11034  11037  11136  11149  11154  11163  11170  11174  11181
                    11493   11499   11699  11706  11712  11715  11777  11790  11899  11910  11930  11935
                    14804   14816   15343  16047  16742  16755  16782  16852  16883  16950  17517  18241
                    18245   18279   18291  18321  18685  18688  18711  18733  18765  18769  18772  18910
                    19102   19105   19109  19113  19117  19121  19125  19129  19193  19196  19200  19204
                    19208   19212   19216  19220  19325  19421  19424  19427  19489  19590  19595  19728
                    20419   20487   20757  21060  21286  21362  21477  21559  22058  22946  22955  23509
                    23699   24450   24558  25969  25973  25994  26006  26009  26014  26017  26021  26029
                    26032   26361   26364  26370  26373  29754  29758  30019  34566  35645  35655  36813
                    36942   37887   38811  39457  39463  39475  40991  41001  41027  41038  41065  41076
                    41103   41114   41140  41145  41150  41176  41186  41233  41260  41271  41333  41359
                    41367   41381   41425  41486  41494  41505  41525  41533  41544  41600  41608  41777
```

Cross Reference Listing - Macro Names

```
MEMSCR_M[].ANDNOT.(R[].RL.24)   4058 #  35546
MEMSCR_M[].RL.24                4059 #  6651   6769   7398   7426   35450
MEMSCR_M[] ZEXT(MB)             4060 #
MEMSCR_RTEMP7_M[]-ZLIT0[]       4061 #
MEMSCR_R[]                      4062 #  6277   6987   37733
MEMSCR_R[] M[].RL.24            4063 #  35579
MEMSCR_ZLIT24[]                 4064 #  5523   5605   37807  37813  37856  47190  47230
MICRO VECTOR?                   5160 #  37409  38931  39172
MM.ALLOW.INT?                   5162 #  23000  23060  23079  23145  23198  23296  23326  23436  23540  23562  23817
                                23834   24196  34436  35521  35562  35589  39919  39975  40034  40045  40261  40376
                                40386
MM.NOINT?                       5163 #  7853   7857   29656  29707
MOPZERO (MUL1 XOR MUL2)?        5165 #  14828  14980
MOPZERO(FLAG1)?                 5166 #  12845  13064  13982  14425  15806  15812  15856  15861  15869
MTEMP0_(MDR.RR.P).AND.R[]       4066 #  29872  29881  29891  29895
MTEMP0_D_0 Q_SEXT(MDR)          4067 #  9300
MTEMP0_D_MDR-ZLIT0[]            4068 #
MTEMP0_MDR+1                    4069 #  19388
MTEMP0_MDR+Q                    4070 #  19290  19296
MTEMP0_MDR+R[]                  4071 #
MTEMP0_MDR+R[]+ALKC             4072 #  14247
MTEMP0_MDR-ZLIT0[]              4073 #
MTEMP0_MDR.AND.ZLIT8[]          4074 #  41647  41668
MTEMP0_MDR.AND.R[]              4075 #  29906
MTEMP0_MDR.AND.ZLIT0[]          4076 #
MTEMP0_MDR.NOTAND.R[]           4077 #  41708
MTEMP0_MDR.OR.R[]               4078 #  41712
MTEMP0_MDR.OR.ZLIT0[]           4079 #  41672
MTEMP0_MDR.OR.ZLIT16[]          4080 #  41651
MTEMP0_MDR.OR.ZLIT8[]           4081 #  45673
MTEMP0_MDR_(MDR+ZLIT0[]).SL.1   4082 #
MTEMP0_PC-R[]                   4083 #  28407  28974
MTEMP0_PC.ANDNOT.R[]            4084 #  41718
MTEMP0_PC.OR.R[]                4085 #  41722
MTEMP0_RBSP                     4089 #
MTEMP0_R[] Q_MDR                4086 #  44985  45331  45502  45874
MTEMP0_R[]-PCBACK               4087 #  39043
MTEMP0_R[].RR.24 Q_PCBACK       4088 #  27768  28846  29120  32034  32692
MTEMP0_SEXT(MDR).XOR.Q          4090 #  8311   8315
MTEMP0_SEXT(XB)+R[] PC_PC+2     4091 #  15406
MTEMP0_SEXT(XB)+R[] PC_PC+4     4093 #  44959  45088
MTEMP0_SEXT(XB)+R[] PC_PC+1     4095 #  44954  45083
MTEMP0_UNPACK(MDR R[])          4096 #  12154  12165  12977  12988
MTEMP0_VA+ZLIT0[]               4097 #  20742
MTEMP0_VA-R[]                   4098 #  28953
MTEMP0_XB PC_PC+4               4099 #
MTEMP0_ZEXT(M[])                4100 #  41251  41276
MTEMP0_ZEXT(XB) PC_PC+1         4101 #
MUL1(FLAG2) XOR MUL2(FLAG3)?    5168 #  12864  13090  13884  14436  15776  16583  16623
MUL2(FLAG3)?                    5169 #  15818
MULFAST+ CAND IN R[]            3739 #  9168   9173   9178   9183   9189   9194   9198   9202   12849  14207  14218
                                14233   14253  14670  15781
MULFAST- CAND IN R[]            3740 #  9207   9212   9217   9222   9228   9233   9237   9241
M[] BIT24?                      5171 #  38152
M[].EQ.-1?                      5172 #  9829
```

```
M[]<31-16>.EQ.FFFF?                 5173 #  35363
M[]_(MB Q).RL.1                     4104 #  11555   11644   15000
M[]_(MB Q).SL.1                     4105 #  14076   14097   15836   15841
M[]_(MB R[]).RR.4                   4106 #  7792    7812    35433
M[]_(MB R[]).RR.9                   4107 #
M[]_(MB R[]).RR.P                   4108 #  11373   12545   12590   12615   15136   15146   15169   15212   47666   47670   47736
                                    47776   47844   47848   47851   47991   48036
M[]_(MB R[]).RR.PS                  4109 #  18486   18757
M[]_(MB R[]).RR.S                   4110 #  18746
M[]_(MB R[]).XZ                     4111 #  18501   18516
M[]_(MB+ALKC).BCD                   4112 #  27229   27233   31682   31685   31691
M[]_(MB+R[]).BCD                    4113 #  26699   26714   26735   26740   27160   27164   27168   27172   27446   28461   29519
                                    31678
M[]_(MB+R[]).RL.1                   4114 #  21330   36307
M[]_(MB+R[]).RR.1                   4115 #
M[]_(MB+R[]).SL.1                   4116 #  14474   15922   29899
M[]_(MB+R[]).SR.1                   4117 #  16581
M[]_(MB+R[]+ALKC).BCD               4118 #  26757   26760   26763   27206   27216   27226   27449   28522   28524   28527   29514
                                    29529
M[]_(MB+R[]+ALKC).SL.1 ALU<0>_0     4119 #
M[]_(MB+R[]+ALKC).SL.1 ALU<0>_1     4120 #
M[]_(MB+ZLIT24[]).BCD               4121 #  29511
M[]_(MB-R[]).BCD                    4122 #  26704   26709   26730   26745   28457
M[]_(MB-R[]).SL.1                   4123 #  14455   14470
M[]_(MB-R[]-ALKC).BCD               4124 #  26808   26811   26814   28469   28484   28488
M[]_(MB-R[]-ALKC).SL.1 ALU<0>_0     4125 #
M[]_(MB-R[]-ALKC).SL.1 ALU<0>_1     4126 #
M[]_(MB.RR.4).NOT                   4127 #  36068
M[]_(R[] MB).RL.4                   4128 #  7895    79[illegible]  28745   30913   30918   32274   32277   32280   32285
M[]_(R[] MB).RL.P                   4129 #  31591   315[illegible]  31598   31603
M[]_(R[] MB).RL.PS                  4130 #  18761
M[]_(R[]+PL).SR.1                   4131 #  29776
M[]_(R[].ASL.1)-MB                  4132 #  28240
M[]_(R[].ASL.SIZE)                  4133 #  45233
M[]_-1                              4134 #  5973    21259   24774   25075
M[]_-MB                             4135 #  11367   12368   12738   12741   29473   29937   31508
M[]_-R[]                            4136 #  18202   21819   21823
M[]_0                               4137 #  8173    14728   16588   22450   22454   23133   23283   24741   24744   25022   27055
                                    30036   41480   41519   47634   47684   47963   47969   47978   48043
M[]_0+ZLITO[]                       4138 #
M[]_ASTLVL                          4139 #  35782
M[]_ATCR                            4140 #  38528
M[]_CC_ZLITO[]                      4141 #  27665   31441
M[]_CONREGS                         4142 #  6135    7954    8018    8040    8045    36087   38205   38219
M[]_CONX(1)                         4143 #  38614   38733
M[]_CONX(2)                         4144 #  6166    7444    7488    8194    38539   38620
M[]_CONX(4)                         4145 #  7510
M[]_CONX.SIZ                        4146 #  47082
M[]_CVTNP(MB)                       4147 #  30505   30741
M[]_D                               4148 #  7876    12863   14676   15792   46720
M[]_D+(R[].RR.16)                   4149 #  36072
M[]_D+ALKC                          4150 #  14228
M[]_D+R[]                           4151 #  22969
M[]_D+R[]+ALKC                      4152 #  13881   14241   16491
M[]_D+ZLIT12[]                      4153 #  7065    7077    7089    7101
```

Cross Reference Listing - Macro Names

```
M[]_D+ZLIT8[]                4154 #  5398    5408    5590
M[]_D-ZLIT0[]                4155 #  20200
M[]_D-ZLIT12[]               4156 #
M[]_D-ZLIT8[]                4157 #  7062    7074    7086    7098
M[]_D.AND.ZLIT0[]            4158 #  35328
M[]_D.AND.ZLIT8[]            4159 #  24782
M[]_D.ANDNOT.ZLIT0[]         4160 #
M[]_D.OR.(MB.RR.16)          4161 #  35334   47814
M[]_D.OR.PACK(MB R[])        4162 #  11824   11888   12575
M[]_D.OR.ZLIT8[]             4163 #  36151   36156
M[]_D_(MB R[]).RR.P          4164 #
M[]_D_(MB+Q).SL.1 Q<0>_1     4165 #  46286
M[]_D_D.ANDNOT.ZLIT0[]       4166 #  20127   20138
M[]_D_Q                      4168 #  18651
M[]_D_Q Q_MB                 4169 #  15684   35038
M[]_D_Q_0                    4170 #
M[]_D_Q_D+Q+1                4171 #  46276
M[]_D_REM                    4172 #  9453    9464    9470    42118   42124   42133
M[]_D_R[]                    4173 #  9309    20110   20116   25056   33813   34758
M[]_D_SEXT(MB)               4167 #  42107
M[]_D_ZLIT0[]                4174 #  14835   47493   47508
M[]_D_ZLIT12[]               4175 #  14840   47498   47503
M[]_D_ZLIT16[]               4176 #
M[]_FLAGS                    4177 #  23743   23747   23751   23791   23796   23857   23861   23865   24660   29765   31309
                             39033
M[]_FPTCR                    4178 #  38533
M[]_MB+(R[].ASL.1)           4179 #  27820
M[]_MB+(R[].ASL.2)           4180 #  29902
M[]_MB+(R[].RR.24)           4181 #
M[]_MB+(R[].SL.2)            4182 #  33119
M[]_MB+1                     4183 #  6585    6603    7658    7670    28510   28582   28588   30193
M[]_MB+PL                    4184 #  12554   47749
M[]_MB+Q                     4185 #  14223
M[]_MB+R[]                   4186 #  7978    11351   12843   13870   14825   14976   15816   15822   16472   24880   28248
M[]_MB+R[]+ALKC              4187 #  11344   12598   12626   47744   47856   47859   47862   47865
M[]_MB+ZLIT0[]               4188 #  7110    7689    7828    7832    7911    7915    11326   11340   11798   12549   12594
                             13084   14441   14710   28506   28601   29559   30451   34197   34318   47982
M[]_MB+ZLIT12[]              4189 #
M[]_MB+ZLIT16[]              4190 #  34405
M[]_MB+ZLIT24[]              4191 #  37431
M[]_MB+ZLIT8[]               4192 #  37867   38766   38927   39157   47740
M[]_MB-1                     4193 #  5635    7203    7223    28322   31533   31538   31553
M[]_MB-CONX(1)               4194 #  6607    7795    7814    7923
M[]_MB-CONX(2)               4195 #  31548   31574
M[]_MB-R[]                   4196 #  22536   24704   31905   31908
M[]_MB-R[]-ALKC              4197 #
M[]_MB-ZLIT0[]               4198 #  5333    5539    5681    5784    12236   12619   12857   13478   13585   13622   13875
                             14687   15005   15012   15710   15827   15874   15879   15903   16482   16524   16666   21213
                             28245   28284   28327   28532   28537   28669   30599   30785   32648   32658   32663   32667
                             33902   34003   34071   34088   34149   34259   35566   36051   47780
M[]_MB-ZLIT12[]              4199 #
M[]_MB-ZLIT16[]              4200 #  35986
M[]_MB-ZLIT8[]               4201 #  34171   47935   47942
M[]_MB.AND.CONX(1)           4202 #
M[]_MB.AND.OLIT0[]           4203 #  15811   16622   20465   42044
```

Cross Reference Listing - Macro Names

```
M[]_MB.AND.OLIT16[]      4204 #  37282  37311  37316  37321
M[]_MB.AND.OLIT24[]      4205 #  17634  36353  37194  47651
M[]_MB.AND.R[]           4206 #  18207  35967
M[]_MB.AND.ZLITO[]       4207 #  8057   8062   8072   8143   15389  16290  17608  20810  21324  22785  23204
                         23332   24647  24929  25203  25283  29112  29133  29137  29141  29176  29249  29268
                         29849   31227  31230  31241  32044  32099  32131  32172  33668  33672  33676  34167
                         34301   34393  34440  34458  34498  34658  34679  34701  35030  35268  36063
M[]_MB.AND.ZLIT16[]      4208 #  23030  23034  23038  23095  23167  23171  23175  24952  25243  35314  38173
                         39038
M[]_MB.AND.ZLIT24[]      4209 #  47281
M[]_MB.AND.ZLIT8[]       4210 #  24680  25025  25165  46874
M[]_MB.ANDNOT.CONX(1)    4211 #  7401   7429   7450   7472
M[]_MB.ANDNOT.CONX(2)    4212 #  7495   7517   7539
M[]_MB.ANDNOT.CONX(4)    4213 #  6516
M[]_MB.ANDNOT.OLIT16[]   4214 #  38850
M[]_MB.ANDNOT.R[]        4215 #  18226  21358
M[]_MB.ANDNOT.ZLITO[]    4216 #  6544   7564   7753   7789   7809   29599  33037  34560
M[]_MB.ANDNOT.ZLIT12[]   4217 #  20778  20786
M[]_MB.ANDNOT.ZLIT16[]   4218 #  29589  32952  38222
M[]_MB.ANDNOT.ZLIT24[]   4219 #  7187   29584  29604  31688  32573  32903  32983  35176  36060  36732
M[]_MB.ANDNOT.ZLIT28[]   4220 #  20428  37787  40237  40605
M[]_MB.ANDNOT.ZLIT8[]    4221 #  7591   29594
M[]_MB.ANDNOT.ZLITPL[]   4222 #  20804  21318  37486
M[]_MB.ASR.P             4223 #  12891  18334  35961
M[]_MB.BCDSWP            4224 #  28741  28749  32197  32214  32234  32251  32270  32288  33182  33190  33198
                         33215   33218  33221  33229  33232  33235
M[]_MB.CLR1B             4225 #  33014
M[]_MB.CLR2B             4226 #  32490
M[]_MB.OR.(MB.RL.16)     4227 #  22764
M[]_MB.OR.(MB.RL.8)      4228 #  22761
M[]_MB.OR.(R[].RL.8)     4232 #  34462  34501
M[]_MB.OR.(R[].RR.16)    4229 #  27989
M[]_MB.OR.(R[].RR.24)    4230 #  28014
M[]_MB.OR.(R[].RR.8)     4231 #  33155
M[]_MB.OR.CONX(1)        4233 #  7386   7413   7441   7462
M[]_MB.OR.CONX(2)        4234 #  7485   7507   7529
M[]_MB.OR.CONX(4)        4235 #  6512
M[]_MB.OR.R[]            4236 #  24944  33073  33632  41408
M[]_MB.OR.ZLITO[]        4237 #  6541   7560   7749   20490  20494  20498  20791  24664  24933  28783  33998
                         34177   34273  38370  38379  39385  40149  40153
M[]_MB.OR.ZLIT12[]       4238 #  17592
M[]_MB.OR.ZLIT16[]       4239 #  36908  37204  37384  38855
M[]_MB.OR.ZLIT24[]       4240 #  25676  25680  26789  26793  26847  26852  27374  27378  29534  29538  31703
                         31707   35372  37270  37274  37278  47646
M[]_MB.OR.ZLIT28[]       4241 #  20437  37722  40225  40597  47654
M[]_MB.OR.ZLIT8[]        4242 #  7581   33541  33545  33916  34028  34075  34092  34132  34153  34157  34240
                         34263   34267  34294
M[]_MB.OR.ZLITPL[]       4243 #  35287
M[]_MB.RL.1              4244 #  29499  46822  46838
M[]_MB.RL.24             4245 #  36299
M[]_MB.RL.4              4246 #  35954
M[]_MB.RL.8              4247 #  6580   6600   35951  36034
M[]_MB.RR.16             4248 #  6617   8050   24973  36015  36146  39024  47629  47639  47679  47689  47797
                         47886   47889  47892
M[]_MB.RR.24             4249 #  33001  33006
```

Cross Reference Listing - Macro Names

```
M[]_MB.RR.4                4250 #  36043
M[]_MB.RR.8                4251 #  24692   32155
M[]_MB.RR.P                4252 #  29494
M[]_MB.SR.1                4253 #  15305   15855   15860   27454   29779   34547
M[]_MB.XOR.ZLIT12[]        4254 #  10867   10878
M[]_MEMSCR                 4255 #  35543   35948   36096   36120   37690   37708   37804   37827   37841   38033   38149
                           47278
M[]_OLIT0[]                4256 #  6288    6302    6316    6330    6344    6358    6372    6400    6414
M[]_OLIT16[]               4257 #  28493   36101   39120   48096   48099
M[]_OLIT24[]               4258 #  24477   25367   36106
M[]_OLIT8[]                4259 #
M[]_PACK(MB R[])           4260 #  11880
M[]_PL                     4261 #  11379   29485   35293   37512
M[]_PL_Q                   4262 #  18363   18367   18371   18671   18678
M[]_PS[                    4263 #  12633   15385   20155   20950   21350   21411   30029   33559   34704   35778   36248
                           36345   36350   36618   36796   36895   37181   37187   37334   40857   41362   41384   42041
                           42171   42507   42670   47570
M[]_Q                      4264 #  8291    8909    9018    9022    9027    9323    9381    10991   10995   11001   11233
                           11243   11334   11440   11570   11629   11977   12046   12160   12232   12833   13052   13089
                           13156   13232   13453   13618   13766   13785   14082   14309   14328   14493   14639   14940
                           15238   15299   15304   16440   16447   16505   16986   21005   21030   22502   22561   24847
                           24988   28717   28729   28737   29524   33487   34964   34974   35247   35265   35398   39222
                           40212   40221   46495
M[]_Q Q_D                  4265 #  9669    19278   36625
M[]_Q Q_MB PL_[]           4266 #
M[]_Q_D+R[]+ALKC           4267 #  14991
M[]_Q_D_R[]                4268 #  24307
M[]_Q_FLAGS_R[]            4269 #  25015   25371
M[]_Q_MB.ASR.P             4270 #  12887
M[]_Q_MB.XOR.R[]           4271 #  17521
M[]_Q_R[]                  4272 #  6260    17629
M[]_Q_UNPACK(MB R[])       4273 #  12957   12967
M[]_Q_ZLITO[]              4274 #  46478
M[]_R[]                    4275 #  5438    5448    5754    6678    6743    6811    6864    6956    7192    7215    7238
                           7287    7392    7396    7423    7447    7469    7492    7514    7536    7804    7823    8097
                           8103    13609   15471   15770   16225   16238   16255   16259   16414   16425   16428   16456
                           17619   18378   18382   19575   20186   21099   21201   23137   23288   23684   23936   24531
                           24701   24735   24757   24828   24992   25080   25302   25307   25379   25405   25639   25643
                           26130   26150   26223   26237   26240   26274   26277   26509   26515   27117   27121   27125
                           27129   27423   27441   27465   27919   27949   28125   28128   28131   28134   28302   28306
                           29083   29106   29159   29272   29412   29462   29469   29481   29623   29627   29814   29834
                           29924   30210   30567   31039   31043   31068   31128   31172   31218   31233   31235   31460
                           31482   31493   31887   32054   32058   32071   32089   32096   33663   33858   33862   33898
                           33907   34067   34084   34192   34314   34666   34685   34692   34754   34872   34876   35353
                           35644   35770   35892   35991   35996   36026   36076   36131   36267   36791   36919   36965
                           37199   37230   37235   37240   37265   37287   37984   38275   39162   39381   39441   39445
                           39449   39452   41301   41952   42076   42426   42643   45114   45442   45589   45648   45702
                           45758   46364   46521   46560   46980   48048
M[]_R[] Q_MB               4276 #  18459   18464   18469
M[]_R[] Q_ZEXT(MB)         4277 #
M[]_R[]+0                  4278 #
M[]_R[]+1                  4279 #  28347
M[]_R[]+Q                  4280 #  26255   26283
M[]_R[]-0                  4281 #  34253   34349
M[]_R[]-1                  4282 #
```

```
M[]_R[]-D-ALKC                  4283 #  41990   42002   42007
M[]_R[]-MB                      4284 #  13061   14429
M[]_R[]-MB-ALKC                 4285 #
M[]_R[]-Q                       4286 #  21719   41973   41979   41995   42489
M[]_R[].ASL.1                   4287 #  45677
M[]_R[].ASL.P                   4288 #  14512
M[]_R[].NOT                     4289 #  46983
M[]_R[].OR.((MB RB).RR.4)       4290 #  30826
M[]_R[].OR.((RB MB).RL.4)       4291 #  7655    7667
M[]_R[].OR.D                    4292 #
M[]_R[].RR.16                   4293 #  16287   17601   24675   24747   24785   25061   25210   25276   29100   29109   29346
                                31238   31881   34654   34676   35971   37143
M[]_R[].RR.24                   4294 #  6620    22781   23680   23932   25288   28151   29180   29223   29253   29280   29424
                                31221   31871   34736   34740   34744   39074
M[]_R[].RR.8                    4295 #  25196   27937   28703   29103   29334   29391   31214   31578   32040   32115   32139
                                32143   33715   33731   34309   34710
M[]_R[].RR.PS                   4296 #  18482
M[]_R[].RR.SIZ                  4297 #  41404
M[]_R[].SL.1                    4298 #  27813
M[]_R[].SR.1                    4299 #  33492
M[]_R[].XZ                      4300 #  18497   18512
M[]_SEXT(MB)                    4301 #  29129   29145   41305
M[]_SEXT(MB).XOR.Q              4302 #  8296    8300
M[]_STEPC                       4303 #  20800   21314   29846
M[]_TCSR.IICR                   4304 #  8035    35050   35054   35801   35809   38816   38897   38905
M[]_TOYOS.TOYCN                 4305 #  36010
M[]_TRAR_ZLIT16[]               4306 #
M[]_TU58REGS                    4307 #  5491    35387   36081   38084   38098
M[]_UNPACK(MB R[])              4308 #  11793   12061   12069   12135   12145   13371   13981
M[]_VA_R[]-1                    4309 #  22991
M[]_WB                          4310 #  36911
M[]_ZEXT(MB)                    4311 #  8320    24752   24768   24824   24936   25035   25069   25318   25382   29154   29184
                                29234   29257   29284   29421   29427
M[]_ZEXT(MB)+R[]                4312 #  8087    32111   32118
M[]_ZEXT(MB)-R[]                4313 #  33992
M[]_ZEXT(MB).SR.1               4314 #  29639
M[]_ZLITO[]                     4315 #  5365    5384    5402    5471    5476    5503    5534    5617    5662    5731    5977
                                6127    6148    6155    6159    6174    6224    6255    6264    6292    6306    6320    6334
                                6348    6362    6376    6386    6390    6404    6418    6432    6486    6502    6527    6535
                                6556    6612    6752    6803    6825    6831    6857    6931    7176    7263    7267    7275
                                7279    7283    7309    7382    7404    7409    7419    7432    7437    7453    7458    7465
                                7475    7498    7503    7520    7525    7532    7542    7547    7556    7568    7577    7594
                                7651    7663    7705    7756    7761    7768    7860    7871    7985    8093    8186    8190
                                8205    15689   15846   16198   16230   16435   17117   17122   17562   20965   21115   21136
                                21910   21968   22637   22677   23004   23078   23128   23237   23278   23688   24422   24471
                                24540   24760   24805   24868   25040   25109   25119   25299   25648   25790   26293   26581
                                27013   27017   27022   27026   27043   27051   27101   27731   27774   27778   27800   27817
                                27868   27872   27876   27911   27915   27983   28007   28048   28064   28106   28117   28143
                                28147   28182   28187   28213   28311   28341   28417   28427   28453   28769   28778   28855
                                29074   29089   29246   29491   29818   30104   30201   30295   30351   30456   30460   30480
                                30491   30518   31504   32080   32122   32162   32420   32423   32426   32438   32459   32478
                                32494   32545   32564   32567   32570   32590   32594   32598   32602   32607   32611   32615
                                32624   32746   33128   33150   33482   33527   34362   34682   35118   35570   35574   35797
                                35813   36111   36115   36287   36335   37246   37251   37426   37628   37634   37640   37646
                                37726   38270   38345   38351   38384   38427   38546   38551   38609   38626   38631   38637
```

Cross Reference Listing - Macro Names

```
R[].SIZ_0                          4529 #  8521   40989  40999  41005  41566
R[].SIZ_D                          4530 #  34011  42026  42032  42163  42838  42984
R[].SIZ_FLAGS                      4531 #  25171  28959  31169  31864  33244
R[].SIZ_FPA                        4532 #  16740  16753  16767  16808  16323
R[].SIZ_M[]                        4533 #  6590   8356   8440   8985   9339   11478  11485  11667  11696  24640  24948
                                   24969  24997  25102  25401  33782  35359  35982  36018  36160  41332  41451  42014
                                   42019  42388  42447  42619  42629
R[].SIZ_M[]+1                      4534 #  8619   41062  41073  41080
R[].SIZ_M[]+Q                      4535 #  8595   8601
R[].SIZ_M[]+Q+PSLC                 4536 #  41599  41607  41626
R[].SIZ_M[]+RB                     4537 #  8589   8635   19341  41858
R[].SIZ_M[]+RB+PSLC                4538 #  41485  41493
R[].SIZ_M[]-1                      4539 #  8708   41024  41035  41042
R[].SIZ_M[]-Q                      4540 #  8684   8690   8930   8941   9318
R[].SIZ_M[].AND.RB                 4541 #  29092
R[].SIZ_M[].ANDNOT.Q               4542 #  8856   8864
R[].SIZ_M[].ASR.-P                 4543 #  41327
R[].SIZ_M[].BCDSWP                 4544 #  36056
R[].SIZ_M[].NOT                    4545 #  8367
R[].SIZ_M[].NOTAND.RB              4546 #  8850   42313
R[].SIZ_M[].OR.Q                   4547 #  8812   8819   8913   9327   9505   14952  42037
R[].SIZ_M[].OR.RB                  4548 #  8806   16073  42247
R[].SIZ_M[].RR.8                   4549 #
R[].SIZ_M[].RR.F                   4550 #  41231  41655
R[].SIZ_M[]..R.SIZ                 4551 #  41447
R[].SIZ_M[].SL.1                   4552 #  41257  41268  41279
R[].SIZ_M[].XOR.Q                  4553 #  8768   8775   42209  42220
R[].SIZ_M[].XOR.RB                 4554 #  8762
R[].SIZ_Q_Q_D                      4556 #  8350   8435   9335   41314  41819  41829  42147
R[].SIZ_Q-RB-1                     4555 #  41174  41184
R[].SIZ_Q_D_M[].ASL.P              4557 #  41321
R[].SIZ_Q_M[]                      4558 #  8306   16029  16637
R[].SIZ_Q_M[].RR.SIZ               4559 #  41761  41770
R[].SIZ_RB+1                       4560 #  11807  38832  38844  39002  39006  39053  39060  39096  39103
R[].SIZ_RB+CONX(2)                 4561 #  42845  42991  43130  43166
R[].SIZ_RB+D                       4563 #  43125
R[].SIZ_RB-1                       4565 #  15799  15850  15916  16211  16267  16510  16548  16567  16593  16994  17030
                                   17052  17242  17303  17491  42477
R[].SIZ_RB-CONX(2)                 4562 #  43066
R[].SIZ_RB-D                       4564 #  43157
R[].SIZ_RB-M[]                     4566 #  8678   41890
R[].SIZ_RB-M[]-PSLC                4567 #  41524  41532
R[].SIZ_RB.SL.1                    4568 #
R[]_((M[] RB).RR.4.AND.MB).SL.1    4570 #  27498
R[]_(M[] RB).RR.4                  4571 #  17546  32200  32203  32206  32210  32237  32240  32243  32247
R[]_(M[] RB).RR.9                  4572 #  12603  36128
R[]_(M[] RB).RR.P                  4573 #  13688  16578  31617  31622  31626  31629  31634  47873  47877  47881
R[]_(M[] RB).RR.PS                 4574 #  18430  18729
R[]_(M[] RB).RR.S                  4575 #  18707  47788  47793
R[]_(M[] RB).XZ                    4576 #  18442  18454
R[]_(M[]+CONX(2)).SR.1             4577 #  27737
R[]_(M[]+RB).BCD                   4578 #  27431
R[]_(M[]+RB+ALKC).BCD              4579 #  27435
R[]_(M[]-RB).BCD                   4580 #  26827
R[]_(M[]-RB-ALKC).BCD              4581 #  26835  26838  26841
```

Cross Reference Listing - Macro Names

```
R[]_(M[].AND.RB).RL.1          4582 #  36142
R[]_(RB M[]).RL.4              4583 #  28558   28573
R[]_(RB M[]).RL.P              4584 #  18750
R[]_(RB+Q).SL.1                4585 #  33115
R[]_(RB-M[]<3-0>)              4586 #  38974
R[]_(RB.OR.M[]<3-0>)           4587 #  35957
R[]_-1                         4588 #  9034    9115
R[]_-D                         4589 #  9526
R[]_-M[]                       4590 #  7207
R[]_-RB                        4591 #
R[]_-ZEXT(M[])                 4592 #
R[]_0                          4593 #  5352    5362    5413    5431    6251    8163    8526    8535    9031    9103    10697
                               10770   10820   10872   11106   11238   11250   11454   11774   12541   12565   12569   12611
                               12645   12982   13360   14519   14695   14850   14964   15118   15140   15156   15197   15215
                               15225   15231   15242   15746   15868   16047   16273   16477   16633   16649   17004   17048
                               17294   17332   17511   17516   22435   22929   22933   22946   23509   24098   25931   25935
                               25953   25957   26345   26353   26357   26512   26819   27140   27417   27420   27720   27922
                               28852   29739   29742   29745   29927   29953   29968   29973   29977   29991   29996   30019
                               30042   30054   30108   30113   30384   30391   30395   30655   30658   30728   31779   34544
                               34565   46187   46348   46357   46360   47762   47766   47772   47839   47908   48000
R[]_8                          4594 #  6814    6843
R[]_CONX(1)                    4595 #
R[]_CONX(2)                    4596 #  27298
R[]_D                          4597 #  9514    9518    9522    9760    14212   15698   18765   21025   21775   21780   21804
                               21878   21899   22725   22952   23497   23699   24450   27071   27075   28166
R[]_D+CONX(1)                  4598 #  6779
R[]_D+CONX(2)                  4599 #  6787
R[]_D+CONX(4)                  4600 #  6795
R[]_D+RB                       4601 #  29930   29934   38393   38437
R[]_D+RB+ALKC                  4602 #  9058
R[]_D-0                        4603 #  22467   22477   22683   22687
R[]_D.NOTAND.RB                4604 #
R[]_D.OR.(M[].RL.8)            4605 #  33835
R[]_D.OR.RB                    4606 #  28397
R[]_D_(D+RB).SL.1 Q<0>_1       4607 #  46226
R[]_D_-D                       4608 #  23943
R[]_D_-M[]                     4609 #  23234
R[]_D_0                        4610 #  14739   15678   23338   23353
R[]_D_D+CONX(4)                4611 #
R[]_D_D+Q+1                    4612 #  46215
R[]_D_D-1                      4613 #  23997   24004   24092
R[]_D_D-CONX(4)                4614 #  24019   24107
R[]_D_M[]                      4615 #  9376    23214   23342   23693
R[]_D_M[]-0                    4616 #
R[]_D_M[]-1                    4617 #  19448
R[]_D_Q Q_D                    4618 #  34999   35008
R[]_D_Q_-T                     4619 #  46466
R[]_D_Q_RNUM_-1                4620 #  46550   46555
R[]_D_RB+1                     4621 #  24189   24238
R[]_D_RB+CONX(4)               4622 #
R[]_D_RB-1                     4623 #  19455   23110   23261   23530   23535   23880   24215   24219   24323
R[]_D_RB-CONX(4)               4624 #  23190   23319   23821
R[]_D_ZEXT(M[])                4625 #  22418   24792   25084   25213   25312   25325
R[]_EXP(M[])                   4626 #  11550   11639   11760   11982   12051   12127   12838   12949   13056   13151   13227
                               13448   13761   13780   13861   14304   14323   14418   14655   14935   14959   15308   15752
```

Cross Reference Listing - Macro Names

		15786	16025	16039	16246	16419	16555					
R[]_FLAGS	4627 #	21983	22024	22050	25725	26871	26880	29208	31740	31748		
R[]_FLAGS_0	4628 #	12881	13073	13866								
R[]_FPA	4629 #	8981	8989	9111	9121	16780	16865	16874	16892	16937	16948	16954
		17100	17253	17299								
R[]_MEMSCR	4630 #	6145	37917	38362	38449	39943	39954					
R[]_M[]	4631 #	5370	5375	5417	5441	5451	5655	6152	6177	6748	6756	6772
		6828	6867	6895	6925	6935	6959	7017	7116	7119	7122	7125	7271
		7290	7294	8121	8403	8415	9071	9098	9494	9720	10692	10702	10751
		10763	10773	10829	11101	11110	11258	11787	11896	11907	11913	11928	11933
		12227	12336	12347	12378	12395	12404	12693	13165	13244	13472	13640	13663
		13771	13796	14038	14043	14314	14339	14434	14508	14720	14748	14757	14777
		14802	14807	14814	14819	14866	15151	15175	15182	15318	15486	15493	15693
		15706	15729	15892	15909	16005	16052	16124	16277	16452	16460	16466	16496
		16520	16538	16543	16628	16910	16922	16973	17025	17110	17155	17164	17232
		17274	17278	17311	17316	17441	17448	17460	17466	17477	17481	17507	17534
		18278	18427	18439	18451	18726	18769	18772	19533	19539	19545	19607	19611
		19615	19619	19623	19627	19631	19635	19701	19715	20160	20164	20414	20462
		20471	20556	20579	20584	20589	20730	20773	21122	21221	21228	21239	21275
		21361	21458	21517	21549	21972	22008	22424	22508	22567	22581	22603	22609
		22623	22704	22745	22955	22985	23007	23210	23230	23493	23500	23669	23674
		23897	23974	24024	24078	24112	24245	24333	24433	24442	24489	24500	24506
		24551	24560	24629	24688	24697	24863	24883	24963	25177	25394	25625	25629
		25800	25817	25821	25827	25832	25837	25842	26095	26123	26125	26153	26176
		26184	26216	26435	26444	26472	26475	26480	26483	26486	26489	26496	26603
		26615	26620	26623	26626	27007	27029	27032	27035	27038	27292	27315	27325
		27328	27668	27692	27696	29166	29218	29360	29402	29409	29754	29758	29981
		29986	30001	30005	30009	30011	30049	30089	30094	30180	30185	30331	30342
		30433	30443	31119	31123	31139	31163	31181	31333	31426	31469	31479	31911
		31941	32093	32176	32514	32525	32723	32727	32730	32733	32737	33500	33514
		33522	33525	33549	33553	34279	34290	34372	34408	34542	34594	34724	34728
		34732	34952	35179	35221	35356	35746	35774	35790	36263	36271	36275	36279
		36283	36327	36474	36491	36558	36575	36705	36718	36736	36739	36763	36767
		36771	36793	36802	36914	36922	37226	37294	37298	37302	37306	37326	37651
		37747	37872	37908	37911	38163	38262	38292	38296	38301	38354	38357	39011
		39066	39070	39078	39090	39338	39403	39407	39411	39415	39419	39462	39474
		39692	39801	39809	39874	39908	39918	39964	39974	39996	40033	40039	40054
		40140	40313	40375	40380	40398	40566	40584	40692	40701	40755	40762	40814
		40821	40862	40873	40901	40929	41926	42252	42285	42318	42379	42541	43103
		45066	45533	46771	46775	46869	47828	47832	47835				
R[]_M[]+(RB.ASL.SIZ)	4632 #	20767										
R[]_M[]+1	4633 #	6889	18221	36331								
R[]_M[]+CONX(1)	4634 #	6817	6903	6922								
R[]_M[]+CONX(2)	4635 #	6910										
R[]_M[]+CONX(4)	4636 #	6917										
R[]_M[]+PL	4637 #	18240										
R[]_M[]+Q	4638 #	9043	14258	30595								
R[]_M[]+Q+1	4639 #	32794										
R[]_M[]+RB	4640 #	5776	7129	11633	11650	12352	12358	13590	13602	13635	14683	16528
		17613	19335	21782	23514	23903	24536	24650	28858	33893	34661	36311	
R[]_M[]+RB+ALKC	4641 #											
R[]_M[]+RB+PSLC	4642 #	8552										
R[]_M[]+SL	4643 #	18244										
R[]_M[]-0	4644 #											
R[]_M[]-1	4645 #	24010	24088	24575	25182	32719	33518	34553	34599	36469	36553	40705

```
                          40766
R[]_M[]-CONX(2)           4646 #
R[]_M[]-CONX(4)           4647 #  20475   21539
R[]_M[]-Q                 4648 #  9048    16128   21040   21597   21673   21686   24557   24718
R[]_M[]-RB                4649 #  21813
R[]_M[].AND.RB            4650 #  22801   29086   33137   39028
R[]_M[].ANDNOT.Q          4651 #  36759
R[]_M[].ASL.P             4652 #  9747    20183   41412
R[]_M[].ASR.-P            4653 #  9686    9741    9763    12410   41418   41433
R[]_M[].ASR.3             4654 #  18395   18659   19047   19060   19070
R[]_M[].ASR.P             4655 #  9773    9781    12389   13706   18290
R[]_M[].BCDSWP            4656 #  27086   28713   28721   28733   29545   29549   41746   41755   41766
R[]_M[].CLR1B             4657 #  35094
R[]_M[].CLR2B             4658 #
R[]_M[].CLR3B             4659 #  28069
R[]_M[].FPLIT             4660 #  17321   17326   45669
R[]_M[].NIBBLE            4661 #  6800    7113    7638    7686    7781    7786    7899    17527
R[]_M[].OR.(RB.RL.16)     4663 #  34623
R[]_M[].OR.(RB.RR.16)     4662 #  24721
R[]_M[].OR.(RB.RR.SIZ)    4664 #  16013
R[]_M[].OR.Q              4665 #  21035   24672
R[]_M[].OR.RB             4666 #  22804   24632   24678   24956   24979   25168   25249   28956   31874   31884   31923
                          31929   31938
R[]_M[].RL.24             4667 #  33503
R[]_M[].RL.9              4668 #  35621   40074   40079   40157   40162   40244   40336   40420   40425   40609   40639
                          40642
R[]_M[].RL.P              4669 #  8496    8510    18702   18754
R[]_M[].RR.16             4670 #  11114   12606   13312   13323   13355   13366   13949   13959   13968   13977   15476
                          24815   27506   27711   27941   32466   32485   34008   42658
R[]_M[].RR.24             4671 #  26607   27494   27945
R[]_M[].RR.8              4672 #  6163    27520   32445   46996
R[]_M[].RR.P              4673 #  18698   18714   34387
R[]_M[].RR.PS             4674 #  18423
R[]_M[].SR.1              4675 #  26135   26168   27682
R[]_M[].XOR.RB            4676 #  17539
R[]_M[].XZ                4677 #  18435   18447   20560   20564   20597   35908   36622   47613   47619   47695
R[]_M[]<3-0>+RB           4678 #  38970
R[]_NOT(M[]).AND.RB       4679 #  23141   23292
R[]_NOT(M[].ASR.-P)       4680 #  41438
R[]_PL_M[]-RB             4681 #  13466
R[]_PL_Q_Q_M[]            4682 #  9665
R[]_PL_Q-M[]              4683 #  13615
R[]_PS[                   4684 #  6694    35650   35659
R[]_Q_Q_D                 4685 #  9725    9730    9834    9839    17595   21010   21015   25620   29457   34960   34977
                          34990   35017   41835
R[]_Q_Q_M[]               4686 #  14052   14062   14073   20103
R[]_Q-M[]                 4687 #  19513   19553   25253
R[]_Q_((M[]+Q).SL.1).OR.1 4688 #  46410   46435   46460
R[]_Q_0                   4689 #  9754    12056   12140   12962   13317   13954   13973   13990   15356
R[]_Q_CONX(1)             4690 #  46394   46419   46444
R[]_Q_CONX(4)             4691 #
R[]_Q_D                   4692 #  22473   22630
R[]_Q_D+RB+ALKC           4693 #
R[]_Q_D_M[]               4694 #  24515
R[]_Q_EXP(M[])            4695 #  13460   16654
```

Cross Reference Listing - Macro Names

```
R[]_Q_M[]                          4696 #  22715   24715
R[]_Q_M[].XZ.MM                    4697 #
R[]_Q_Q-M[]                        4698 #  46528   46533   46538
R[]_Q_RB+Q                         4699 #
R[]_Q_ZEXT(M[])                    4700 #  25098
R[]_RB+1                           4701 #  19394   33959   34161   39117
R[]_RB+CONX(1)                     4702 #  27303
R[]_RB+CONX(2)                     4703 #  27275   42603
R[]_RB+CONX(4)                     4704 #  19738   20418   24644   36891   39341   42652   44454   44544   44640   44729   44913
                                   45038   45136   45245   45564   45800
R[]_RB+CONX.SIZ                    4705 #  44424   44527   44623   44717   44889   45031   45147   45523   45772
R[]_RB+Q                           4706 #  7737    8100    22988   27540
R[]_RB-1                           4707 #  23056   27097   27136   27177   27182   27186   27258   27264   27271   27787   27791
                                   27959   28030   30400   30488   30521   33948   34354   34358   34696
R[]_RB-CONX(2)                     4708 #  27992   28017
R[]_RB-CONX(4)                     4709 #  23144   23295   28078   40778
R[]_RB-CONX.SIZ                    4710 #  40718
R[]_RB-D-ALKC                      4711 #  9066
R[]_RB-M[]                         4712 #  16010   17581   22748   27671   31853   31901   31944   33805   34550   34604   39082
R[]_RB-M[]-PSLC                    4713 #  8726
R[]_RB-Q                           4714 #  7733    24565
R[]_RB.AND.(M[].RR.16)             4715 #  33134
R[]_RB.AND.(M[].RR.8)              4716 #  33131
R[]_RB.ANDNOT.Q                    4717 #  19207   19211   19219
R[]_RB.ASL.1                       4718 #  11458   11816   14645   15161   48027
R[]_RB.ASL.2                       4719 #  11803   37479
R[]_RB.ASL.7                       4720 #  13342   13346   13374   13986
R[]_RB.ASL.SIZ                     4721 #
R[]_RB.OR.(M[].ASL.P)              4722 #  20518
R[]_RB.OR.(M[].RL.16)              4723 #  31178   34607
R[]_RB.OR.(M[].RL.24)              4724 #  22751   25262   34613
R[]_RB.OR.(M[].RL.3)               4725 #  31175   34304
R[]_RB.OR.(M[].RL.9)               4726 #  8150
R[]_RB.OR.(M[].RR.16)              4727 #  27970   28021   28971   33175
R[]_RB.OR.(M[].RR.24)              4728 #  28401   28411   28942   28978   28982   31860   31914   31932   32501   33167
R[]_RB.OR.(M[].RR.8)               4729 #  28217   28945   28962   31947   33164
R[]_RB.OR.(M[].RR.SIZ)             4730 #  17587
R[]_RB.OR.(M[].XZ)                 4731 #  36012
R[]_RB.OR.CONX(1)                  4732 #  29097
R[]_RB.OR.Q                        4733 #  19199   19203   19215
R[]_RB.OR.ZEXT(M[])                4734 #  14649   20219   20233
R[]_RB.RL.16                       4735 #  25161
R[]_RB.RL.8                        4736 #  25001   29734
R[]_RB.RL.P                        4737 #  18738   18742
R[]_RB.RR.24                       4738 #  32888   32893   32906
R[]_RB.RR.8                        4739 #  24983   27674   31166   31857   31920   46915
R[]_RB.RR.P                        4740 #  18663
R[]_RB.XOR.MINUS1                  4741 #  41457
R[]_RB.XZ                          4742 #  39497   39500
R[]_RBSP                           4743 #  22098   37862   38761   38922   39152
R[]_RNUM                           4744 #  9485    14716   14744   14857   14872   16915   16978
R[]_RNUM_MTEMP0.XZ                 4745 #  36258
R[]_SEXT(M[])                      4746 #  8897    8903    9269    9274    9279    9285    9290    9295    11319   11445   11561
                                   25601   26088   26099   26417   26423   26439   31411   31421   31435   32151   36295   41947
                                   42081
```

```
R[]_TB                          4747 #  36125
R[]_VA_M[]                      4748 #  22446
R[]_VA_RB+1                     4749 #  23884
R[]_VA_RB-Q                     4750 #  24572
R[]_WDR                         4751 #  39722  39736
R[]_XB PC_PC+1                  4752 #  23017  23021  23090  23464
R[]_XB PC_PC+4                  4754 #  23154  23158  23782  23844  23848  28060
R[]_XB PC_PC+I                  4756 #  45451  45598  45908
R[]_ZEXT(M[])                   4758 #   8146  15985  17070  17454  17625  17638  20144  20168  20764  21106  21126
                                21131   21243  21248  22514  22525  22574  22615  22652  22697  24446  24456  24876
                                29808   29854  30083  30099  30174  30189  30220  30324  30336  30427  30438  31430
                                31891   32158  33171  33473  34619  34669  36465  36549
R[]_ZEXT(M[]).SL.1              4759 #  19524  41603
R[]_ZEXT(XB) PC_PC+1            4760 #   7198  17499  18970  18975  18980  18985  18990  18995  19000  19004  19658
                                19663   19668  19673  19678  19683  19688  19692  23730  23734  23786  27752  28686
                                28691   30205  30484  30644  30668  30896  30902  30936  30941  30982  30990  31032
                                33146   33571  33623
R[]_ZEXT(XB) PC_PC+2            4762 #  27979  28003  30664  30673  30723  30946
S<3-0>.NE.0?                    5217 #  30356
SAMESIGN(FLAG4)?                5218 #  12348  13591
SET ADD1(FLAG0)                 3756 #  12062  12136  13313  13337  14841  15840  16615  47499  47504
SET ADD2(FLAG1)                 3757 #  12155  13356  15880
SET BOOT(FLAG MMNOINT)          3758 #
SET FLAG0                       3759 #   5647   5654   6540   6670   7138   7552   9153   9158   9558   9564  14499
                                18197   18216  18273  18751  20099  21971  22007  22580  23556  24499  24532  25728
                                25889   26176  26656  27915  28347  28533  29907  30193  32379  32790  33727  33738
                                34178   34274  37217  37483  38615  38621  39123  39369  41434  42911  42933  42953
                                45899   45909  46551  47630
SET FLAG1                       3760 #   7377   7481   9400  12580  12642  14504  15376  17529  17630  18314  20800
                                21763   21853  21993  22532  25629  25862  26619  26771  27177  27182  27186  27257
                                27361   28618  28690  28778  29571  29873  29892  30253  30785  31008  31020  31038
                                31621   31662  31725  32892  32902  32951  32970  33036  33096  35702  37237  41419
                                41980   41991  42027  42427  43052  43074  43092  45915  46556  47680  47809
SET FLAG2                       3761 #   5430   6689   6959   7161   9382  11629  11645  18329  20900  20930  21824
                                21911   22004  24196  25691  25899  25911  25925  25935  26790  26794  26842  27017
                                27343   29644  30658  30887  31119  31508  31719  33530  37115  37120  37130  37136
                                37487   38156  38164  38206  38279  38578  38699  38706  38712  38717  38722  38730
                                42109   42644  42882  42887  47635
SET FLAG3                       3762 #   6386   6599   9613   9618  16655  18371  18678  21054  21063  21165  22720
                                23436   24541  25681  25821  25832  25842  25851  26703  26713  26734  26744  26846
                                27066   27379  28322  28582  28805  29473  30276  30378  31294  31305  31539  33534
                                36332   37345  37422  38552  38556  43023  43028  47560  47685
SET FLAG4                       3763 #
SET FPA(FLAG0)                  3764 #  17096
SET FPD                         3765 #  16006  17461  21968  22435  22622  22666  22716  23703  25620  26123  26472
                                27668   29924  30104  30201  30351  30480  31469  33553
SET GFLOAT(FLAG4)               3766 #
SET MM.NOINT                    3767 #   6481   7860   8404   8527   8558   8732   8915   8932   8982   9044   9049
                                 9329   9341   9501   9506   9726   9736   9742   9749   9755  10752  10766  10776
                                11260   11480  11487  11669  11677  11688  11916  13168  13247  13773  13798  14316
                                14341   14721  14749  14768  14773  14778  14783  15320  15488  16768  16809  16824
                                16867   16876  16932  16938  18721  20176  20570  21116  21137  21233  21254  21454
                                21536   21604  21668  21681  21698  21790  21801  21870  21892  21896  21988  29650
                                29655   29711  29716  29722  29991  30042  32812  36308  36614  36766  36874  37335
                                37834   37838  38258  39086  39821  40870  41500  41539
```

Cross Reference Listing - Macro Names

SET MOPZERO(FLAG1)	3768 #	12963	12984	13955	14965							
SET MUL1(FLAG2)	3769 #	12958	13950									
SET MUL2(FLAG3)	3770 #	12978	13969	14042								
SET OPZERO(FLAG3)	3771 #	12057	12141	13319								
SET OVER(FLAG2)	3772 #	14696	15127	15152	15176	47959	48007	48052				
SET POP1C(FLAG4)	3773 #											
SET READ(FLAG1)	3774 #	16030	16638	17090	17259	17264	17285					
SET REGINT(FLAG1)	3775 #	14873										
SET SAMESIGN(FLAG4)	3776 #	12700	12716									
SET SIGN(FLAG0)	3777 #	10992	11102	11368	11769							
SET STACK FLAG	3778 #	20966	36288	36336	36342	37205	37247	37252	37331	37427	37981	37985
	37990	38034	38271	38283	38659	39648						
SET SUB(FLAG1)	3779 #	13146	13222									
SET UNDER(FLAG3)	3780 #											
SET V	3781 #	8950	11498	11711	23479	29986	32691	35639	38540	41351	41443	42129
	42137	47566										
SET V SIGND CMP?	5219 #											
SET WRITE(FLAG1)	3782 #	10739	10746	10874	11225	11327	11363					
SHIFT SIZE?	5220 #	41306	41409									
SIGN(FLAG0)?	5221 #	11012	11021	11131	11141	11328	11794					
SIGND CMP .NOT.IRO?	5222 #	19399	19405									
SIGND CMP DEF?	5223 #	7107	14689	15007	15015	16119	16269	16511	16550	28328	29869	30585
	30625	33610	33615	33639	34255	34714	35469	35482	46368	46730	46748	
SIGND CMP?	5224 #	7032	9027	9441	9694	11336	11352	12222	12556	13467	15329	15334
	15888	16996	17245	19303	19310	19315	19349	19353	19358	21031	21046	21050
	21143	23469	23694	24314	24343	24485	26172	28248	28279	28323	28583	28589
	29689	29731	30575	30580	30830	31494	33649	34351	35272	38833	40483	40488
	40493	40622	41958	41963	41985	42098	42103	46908	46990	47000	47337	47370
	47751											
SIZE[]	3783 #	5545	5563	5568	5578	5595	5621	5630	5686	5730	5753	5789
	5799	5817	6591	6778	6786	6794	6900	6907	6914	7107	7226	8088
	8147	8240	8245	8260	8265	8278	8292	8297	8301	8307	8312	8316
	8320	8332	8351	8356	8363	8368	8373	8378	8387	8404	8416	8420
	8436	8440	8459	8471	8497	8501	8511	8522	8527	8536	8553	8558
	8564	8590	8596	8602	8606	8620	8636	8644	8653	8679	8685	8691
	8695	8709	8727	8731	8737	8763	8769	8776	8780	8807	8813	8820
	8824	8851	8857	8865	8869	8898	8904	8909	8914	8926	8931	8942
	8946	8982	8986	8992	8995	9023	9027	9038	9043	9048	9059	9067
	9072	9099	9148	9153	9158	9163	9169	9174	9179	9184	9190	9199
	9203	9208	9213	9218	9223	9229	9234	9238	9242	9270	9275	9280
	9286	9291	9296	9301	9305	9314	9319	9323	9328	9335	9340	9382
	9409	9415	9424	9440	9502	9506	9515	9519	9523	9527	9553	9564
	9580	9590	9595	9600	9607	9613	9618	9624	9629	9634	9641	9647
	9652	9687	9693	9706	9720	9726	9731	9737	9743	9748	9755	9764
	9774	9782	9829	9835	9840	10693	10698	10703	10753	10765	10775	10821
	10830	11011	11015	11020	11130	11135	11140	11169	11180	11223	11259	11320
	11335	11351	11446	11465	11471	11479	11486	11562	11657	11668	11676	11687
	11697	11704	11776	11789	11808	11812	11897	11908	11914	11929	11934	12222
	12237	12555	12560	12620	12627	12844	12850	12858	13062	13080	13085	13167
	13246	13467	13674	13678	13772	13797	13871	13876	14208	14219	14234	14254
	14315	14340	14430	14442	14461	14479	14650	14671	14689	14711	14722	14734
	14750	14768	14773	14778	14783	14803	14808	14815	14820	14826	14948	14953
	14977	15007	15014	15078	15094	15319	15329	15334	15487	15494	15503	15509
	15526	15531	15536	15542	15694	15718	15730	15758	15782	15800	15817	15823
	15828	15851	15875	15880	15888	15893	15910	15917	15986	16014	16019	16030

Cross Reference Listing - Macro Names

16033	16074	16119	16212	16217	16268	16474	16487	16497	16511	16533	16539
16549	16568	16593	16609	16616	16638	16642	16741	16754	16769	16782	16810
16825	16850	16866	16875	16893	16932	16938	16949	16955	16995	17009	17014
17019	17031	17040	17053	17071	17081	17089	17100	17243	17260	17265	17284
17303	17455	17477	17492	17535	17556	17588	17626	17638	18197	18216	18236
18240	18244	18249	18273	18279	18287	18291	18314	18321	18329	18401	18711
18721	18733	18915	19075	19097	19109	19113	19117	19121	19125	19129	19188
19249	19291	19297	19303	19310	19315	19336	19342	19349	19353	19358	19389
19395	19400	19404	19449	19455	19519	19525	19538	19544	19558	19565	19571
19578	19589	19594	19644	19727	20109	20115	20145	20154	20169	20197	20208
20220	20229	20234	20239	20248	20257	20265	20274	20283	20292	20300	20309
20318	20327	20335	20344	20424	20433	20441	20454	20468	20535	20552	20580
20585	20590	20611	20620	20630	20640	20650	20660	20669	20679	20689	20699
20709	20718	20760	20765	20768	20815	20820	21026	21031	21036	21041	21046
21050	21107	21127	21132	21142	21147	21209	21218	21244	21249	21263	21362
21446	21457	21464	21476	21482	21527	21530	21538	21550	21554	21558	21593
21611	21618	21624	21659	21674	21689	21705	21711	21726	21734	21779	21797
21805	21812	21861	21877	21884	21900	21907	21920	21965	21999	22003	22015
22049	22057	22092	22417	22428	22462	22467	22477	22497	22513	22524	22537
22545	22556	22573	22596	22614	22632	22653	22660	22683	22687	22698	22709
22716	22917	22973	23045	23102	23182	23258	23306	23310	23473	23520	23693
23703	23756	23767	23803	23812	24190	24201	24205	24215	24219	24226	24232
24240	24257	24263	24323	24376	24447	24457	24480	24485	24560	24640	24753
24769	24793	24825	24877	24937	24949	24968	24998	25036	25070	25085	25099
25103	25172	25214	25313	25319	25326	25382	25402	25600	25649	25713	25791
25796	26089	26100	26172	26212	26345	26419	26425	26441	26582	26895	27727
28171	28249	28278	28296	28323	28328	28418	28584	28588	28593	28618	28622
28632	28809	28824	28828	28831	28836	28849	28865	28959	29093	29129	29145
29155	29163	29184	29207	29227	29234	29257	29276	29284	29339	29350	29369
29397	29421	29427	29555	29566	29570	29574	29619	29640	29689	29692	29702
29706	29731	29748	29809	29841	29855	29869	29910	29930	29937	30000	30045
30083	30099	30117	30125	30130	30174	30189	30214	30221	30243	30257	30266
30272	30285	30289	30324	30336	30356	30373	30377	30387	30427	30438	30450
30531	30535	30575	30585	30609	30618	30625	30709	30746	30830	30852	30856
30850	30864	30924	30958	30963	30973	30978	31072	31169	31410	31423	31431
31436	31495	31533	31538	31744	31864	31892	32037	32081	32112	32119	32123
32152	32159	32407	32411	32434	32442	32455	32463	32474	32489	32498	32843
32847	32925	32929	32964	32969	33011	33030	33041	33172	33245	33466	33474
33499	33513	33610	33615	33638	33649	33783	33952	33993	34012	34255	34350
34378	34435	34494	34620	34670	34714	35271	35360	35469	35482	35629	35632
35655	35983	36019	36057	36160	36255	36296	36360	36465	36470	36488	36517
36522	36549	36554	36572	36631	36684	36776	36812	36870	36881	37353	37360
37368	37993	37997	38001	38005	38009	38039	38168	38418	38482	38486	38571
38662	38833	38845	39002	39006	39054	39061	39097	39104	39334	39338	39796
40028	40042	40199	40371	40383	40483	40488	40493	40555	40622	40687	40714
40719	40751	40810	40853	40990	40995	41000	41006	41026	41031	41037	41044
41064	41069	41075	41082	41102	41107	41113	41120	41139	41144	41149	41155
41175	41180	41185	41192	41206	41214	41225	41232	41251	41259	41264	41270
41276	41281	41305	41314	41322	41328	41333	41341	41405	41413	41419	41424
41448	41452	41486	41489	41494	41499	41504	41525	41528	41533	41538	41543
41567	41572	41591	41600	41604	41608	41613	41619	41627	41656	41727	41762
41771	41776	41782	41786	41801	41806	41821	41825	41831	41836	41853	41859
41868	41885	41891	41900	41916	41922	41931	41948	41953	41958	41963	41984
42015	42020	42028	42033	42038	42082	42098	42102	42108	42148	42158	42163
42210	42216	42221	42242	42248	42258	42275	42281	42291	42308	42314	42323

```
VA_R[]-CONX(4)                    4892 #  20195
VA_R[]-CONX.SIZ                   4893 #
VA_R[]-M[]                        4894 #  21111   33945
VA_R[].ASL.SIZE                   4895 #  44804
VA_R[].SIZ_RB-CONX(2)             4896 #  42776   42782   42925   42931   43072
VA_R[].SIZ_RB-D                   4897 #  42864   43005
VA_R[].XOR.Q                      4898 #
VA_R[]_D-CONX(4)                  4899 #  20190
VA_R[]_M[]                        4900 #  5760    6871    6875    20203   20745   21442   21512   21535   21584   21650   22772
                                  29578   29615   29831   31133   38492
VA_R[]_M[]+CONX(1)                4901 #  20734
VA_R[]_M[]+CONX(2)                4902 #  20738
VA_R[]_M[]+CONX(4)                4903 #  21449   21531
VA_R[]_M[]+RB                     4904 #  18692   21881
VA_R[]_M[]-CONX(4)                4905 #  37678
VA_R[]_M[]-CONX.SIZ               4906 #
VA_R[]_Q_Q_M[]                    4907 #  21762   21852
VA_R[]_RB+T                       4908 #  23115   23119   23265   23269   23540   23544   23888
VA_R[]_RB+CONX(1)                 4909 #  34368   34446   34450   34509
VA_R[]_RB+CONX(4)                 4910 #  5546    21272   23196   23201   23324   23329   23825   23833   23838   36878   39335
                                  40774
VA_R[]_RB+CONX.SIZ                4912 #  40713
VA_R[]_RB-CONX(1)                 4913 #
VA_R[]_RB-CONX(2)                 4914 #  42557
VA_R[]_RB-CONX(4)                 4915 #  5552    8455    8467    19639   19722   20444   20503   20508   20513   21152   21158
                                  36341   36357   36817   37330   37356   37364   37980   37989   38566   38658   38775   39166
                                  39184
VA_R[]_RB-CONX.SIZ                4917 #  44367   44835   45185   45290   45373   45460   45607   45717   45832   45919   46023
VA_SEXT(XB)+R[]_PC_PC+2           4919 #  42792   42942   43197   43207
VA_SEXT(XB)+R[]_PC_PC+1           4921 #  44377   44396   44507   44603   44693   44781   44845   44864   44972   45101   45199
                                  45220   45300   45318   45383   45402   45470   45489   45616   45635   45726   45745   45842
                                  45861   45931   45953   46033   46052
VA_VA+4                           4922 #  5634    5680    5732    5790    5800    8411    8532    9055    9063    9760    9769
                                  9787    9804    11922   14798   15550   15719   15758   16020   16035   16218   16610   16617
                                  16641   16776   16888   17009   17013   17018   17039   17081   17089   17260   17265   17284
                                  18409   18413   18417   18702   18726   20543   20581   20586   20591   20612   20621   20631
                                  20641   20651   20661   20670   20680   20690   20700   20710   20719   20761   21461   21527
                                  21708   28159   29397   30930   36705   36711   36719   36735   36758   36762   36865   36926
                                  36932   36938   39649   39692   39821   39828   40038   40380   45509   45522   45541   45653
                                  45765   45774   45973   46070   46078   46767   46875   47056
VA_VA+4_CLEAR_CACHE               4923 #  5997    35520
VA_XB_PC_PC+4                     4924 #  44382   44850   45205   45305   45388   45475   45621   45731   45847   45936   46038
VA_XB+R[]_PC_PC+4                 4925 #  44592
VA_XB.XOR.R[]_PC_PC+4             4927 #  46882   46889   47008
VA_ZEXT(M[])                      4929 #  42719   43225   43230   47018
VA_ZEXT(M[])+R[]                  4930 #  29162   29275   38167
VA_ZEXT(M[])-R[]                  4931 #
VA_ZEXT(XB)_PC_PC+2               4932 #  42767   42916
VA_ZEXT(XB)+R[]_PC_PC+1           4936 #  24195
VA_ZEXT(XB)-R[]_PC_PC+1           4938 #  46989
VA_ZEXT(XB)-R[]_PC_PC+2           4940 #  46999
VA_ZEXT(XB).XOR.(R[].RR.8)_PC_PC+1        4934 #  47015
VA_ZLITO[]                        4942 #  46165   46814   46831   47139
WB<0>?                            5237 #  5381    5389    5393    5587    7132    26130   26496   27423   27454   27677   28258
                                  29495   29500   29571   30606   30641   30832   30884   30893   31049   31482   31535   31539
```

Cross Reference Listing - Macro Names

```
                        32085   35039   35074   35102   39478   41461   41490   41529   41614   41643   41664   41693
                        41698   41703   41728   41787   41807   41826   41869   41901   42217   42259   42324   42398
                        42411   42427   42435   42439   42443   42465   42530   42558   42598   42626   42644   42667
                        42763   42912   43053   43198   43203   43208   46822   46838
WB<1-0>.NE.0?           5239 #  6586    20111   20117   23074   23115   23248   23265   23669   23884   23969   24809
                        24820   25113   34991   35000   35009   35018   35098   36328   38083   38210
WB<1-0>?                5238 #  6749    6868    7722    21972   24693   27112   27889   32586   33510   34355   34688
                        37228   38293
WB<31-30>?              5242 #  6608    7180    7203    7223    7615    7627    7635    7676    7796    7815    7908
                        8122    8178    9410    9787    9809    9813    11657   11799   12332   12342   12353   13586
                        13623   13652   13683   13707   14678   14992   15352   15793   16204   16263   16487   16492
                        17621   20481   20604   20725   21282   22537   22939   23085   23225   23253   23357   23479
                        23569   23573   23725   23829   23898   23957   24027   24061   24115   24365   24520   24655
                        24802   24925   24959   25046   25091   25357   27097   27136   27178   27183   27187   27257
                        27264   27275   27827   27834   27844   27885   28563   29116   29463   29555   29877   29931
                        29938   30226   30997   31001   31007   31019   31025   32682   33903   33993   34004   34842
                        34846   34850   34900   34904   34968   35043   35047   35059   35071   35087   35140   35148
                        35152   35160   35164   35173   35186   35191   35196   35200   35209   35218   35231   35235
                        35244   35261   35280   35304   35423   35465   35474   35539   35558   35636   35640   35764
                        35787   35795   35806   35830   35846   35869   35877   35881   35889   35901   35944   36038
                        36255   36631   37336   37419   37658   37705   38102   38400   38467   39354   40055   40141
                        40399   40532   40538   41378   42683
WB<5-0>?                5243 #  6483    6773    6896    20225   27145   28038   28160   29124   29415   29575   32049
                        32135   32550   33060   33065   33563   33643   33664   33777   33810   34720   34983   35252
                        35320   35348   35759   35977   37666   37868   38367   38408   38453   38476   38533   38767
                        38823   38928   39016   39049   39147   39158   41561   46402   46427   46452   46487
WB_(D+CONX(1)).SR.1     4944 #  27889
WB_(M[] R[]).RR.4       4945 #  33562
WB_(M[]+ALKC).BCD       4946 #  27195
WB_(M[]+R[]).BCD        4947 #  7799    7818
WB_(M[]-R[]).BCD        4948 #  25884   25890
WB_(M[]-R[]-ALKC).BCD   4949 #  25895   25900   25906   25912   25919   25926
WB_(M[].AND.ZLITO[]).SR 4950 #
WB_(R[] M[]).RL.P       4951 #  18723
WB_(R[] M[]).RL.PS      4952 #  18734
WB_(R[]+Q).SL.1         4953 #  8926    41984
WB_.NOT.(M[].RR.16)     4954 #  35319   35347   35976   38822
WB_.NOT.(M[].RR.P)      4955 #  38087   38209
WB_.NOT.Q               4956 #  35098   35102
WB_D-CONX(1)            4957 #  31673
WB_ATCR                 4958 #  38556
WB_D                    4959 #  8936    8946    28090   32550   32586   41969
WB_D+ALKC               4960 #  8919    41957   41962
WB_D+ZLITO[]            4961 #  33060   33064
WB_D+ZLIT16[]           4962 #  35185
WB_D-OLIT24[]           4963 #  19468   19477   19482
WB_D-PL                 4964 #  18249
WB_D-Q                  4965 #  15333   15508   15530   15535   24485
WB_D-R[]                4966 #  22461
WB_D-ZLITO[]            4967 #  16118   23957   24061   24365   27827   27834   27844
WB_D.AND.ZLITO[]        4968 #  21867
WB_D.AND.ZLIT16[]       4969 #  41354
WB_D.AND.ZLIT24[]       4970 #  35204
WB_D.ANDNOT.ZLITO[]     4971 #  26110   26464   31111   31455
WB_D.XOR.R[]            4972 #  46735
```

Cross Reference Listing - Macro Names

```
WB_D.XOR.ZLIT0[]                4973 #  30498   30734   30780
WB_D.XOR.ZLIT16[]               4974 #
WB_D.XOR.ZLIT24[]               4975 #  39375
WB_EXP(M[])                     4976 #  10683   10738   10745   10808   10909   10914   10976   10981   11092   11121   15313
                                15393   15480   15724
WB_EXP(M[]) Q_ZEXT(MB)          4977 #  11224
WB_FPA                          4978 #  16851   16931
WB_M[]                          4979 #   5388    5392    6647    6861    6990    7180    7694    7698    7721    8278    9415
                                 9424   16515   19348   20603   20724   21715   21771   21994   22939   23225   23357   23479
                                23569   23573   23969   24635   24655   24925   26348   27427   28554   28606   28636   28640
                                28764   29124   29542   29663   29674   29949   30369   30606   30648   30832   30836   30848
                                32049   32135   32857   33046   33082   33809   34020   34841   34845   34849   34886   34967
                                34982   35042   35046   35058   35070   35074   35086   35139   35147   35151   35159   35163
                                35172   35190   35195   35199   35208   35217   35230   35234   35243   35260   35279   35303
                                35422   35464   35473   35539   35557   35616   35635   35639   35759   35763   35786   35794
                                35805   35829   35845   35868   35876   35880   35888   35900   35943   36038   36527   37419
                                37657   37666   37704   37755   37776   38399   38408   38466   38476   39354   39364   39644
                                40390   40854   41138   41143   41148   41154   41346   41423   42666   46566   47352
WB_M[] PL_MB.MSS                4980 #  18230   18235
WB_M[]+63 PL_31 WB<7>EQ0?       5244 #   9770
WB_M[]+CONX(T)                  4981 #
WB_M[]+PSLC                     4982 #  41504
WB_M[]+Q                        4983 #   9055   14238
WB_M[]+Q+PSLC                   4984 #   8563
WB_M[]+R[]+ALKC                 4985 #
WB_M[]+R[]+PSLC                 4986 #  28257
WB_M[]+ZLITO[]                  4987 #  28562   28773   35251
WB_M[]-(MB.CLR2B)               4988 #  32941   32947
WB_M[]-(R[].XZ)                 4989 #  36254   36630
WB_M[]-1                        4990 #  39111
WB_M[]-PL                       4991 #  35271
WB_M[]-PSLC                     4992 #  41547
WB_M[]-Q                        4993 #   5741    9063
WB_M[]-Q-PSLC                   4994 #   8736
WB_M[]-R[]                      4995 #   5806    8265   11010   11135   11180   11464   19314   19398   19557   19593   21045
                                21475   21553   21557   22916   24519   27884   29688   29730   41921   41930
WB_M[]-R[]-ALKC                 4996 #
WB_M[]-ZLITO[]                  4997 #   7106    7614    7626    7634    7675    7907    7973    8075    8079   24707   29115
                                29868   29876   30584   30624   33609   33614   33637   33648   34713   34890   34899   34903
                                35468   35481   37697   39274
WB_M[]-ZLIT8[]                  4998 #  24683   24796   24801   24834   24950   25045   25090
WB_M[].AND.CONX(1)              4999 #   7317    7325    7333    7341
WB_M[].AND.CONX(2)              5000 #   7348    7356    7364
WB_M[].AND.CONX(4)              5001 #   6498
WB_M[].AND.OLITO[]              5002 #
WB_M[].AND.Q                    5003 #   8239   19082   19086   42274
WB_M[].AND.R[]                  5004 #   8244   24763   25064   35168   36722   36748   39346   42280   42290
WB_M[].AND.ZLITO[]              5005 #   6507    7165    7230    7376    7480    8067    8198   12636   14700   15087   19236
                                19241   20349   20354   20359   20364   20369   20374   20378   20383   20388   20607   20617
                                20626   20636   20646   20656   20666   20675   20685   21603   21664   21697   21786   21888
                                21951   21978   22011   22020   27331   27796   28476   28499   28597   28792   28796   29710
                                29715   30247   31012   31061   31587   31649   32628   32638   32686   32915   32919   33026
                                34145   36478   36495   36562   36579   42694   47573
WB_M[].AND.ZLIT12[]             5006 #  11768   11780   20175
WB_M[].AND.ZLIT16[]             5007 #   5499    5715    5823    7962    8030   32644   32706   32937   36904
```

```
WB_M[].AND.ZLIT20[]          5008 #
WB_M[].AND.ZLIT24[]          5009 #  20539   20750   20954   32654   32670   32991   32996   37693   37718   37736   37743
                             37830   37844
WB_M[].AND.ZLIT4[]           5010 #  20393   20398   20403   20408   20695   20705   20715
WB_M[].AND.ZLIT8[]           5011 #  7372    11811   12559   14733   15071   20572   21384   21394   32634   32701   42113
                             47757
WB_M[].ANDNOT.ZLIT0[]        5012 #  15996   16990   17237   19153   25611   29821   34837   34863
WB_M[].ANDNOT.ZLIT24[]       5013 #  29659   29669   32861   32878   32883   33051
WB_M[].ASR.-P                5014 #  9702    9712    9795
WB_M[].ASR.P                 5015 #  35213
WB_M[].BCDSWP                5016 #  32535   32554
WB_M[].CLR2B                 5017 #  32978
WB_M[].OR.R[]                5018 #  9816    9820    33077
WB_M[].RR.16                 5019 #  32963   32968
WB_M[].RR.24                 5020 #
WB_M[].RR.4                  5021 #  6482
WB_M[].RR.8                  5022 #  39015   39048   39146
WB_M[].RR.P                  5023 #  38101   41322
WB_M[].VPN-R[]               5024 #  40482   40487   40492   40621
WB_M[].XOR.Q                 5025 #  5693    21958   30689   30695   46593   47375
WB_M[].XOR.Q Q_(Q.SL,1).OR.1 5026 #  46482
WB_M[].XOR.Q Q_Q.RL.1        5027 #  46572   46579   46586
WB_M[].XOR.R[]               5028 #  30699   36030   46604   46611   46618   46625   46638   46645   46651   46658   46671
                             46678   46685   46692
WB_M[].XOR.ZLIT0[]           5029 #  6228    6280    6284    6298    6312    6326    6340    6354    6368    6382    6396
                             6410    6424    6463    6470    6477    7313    7321    7329    7337    7344    7352    7360
                             7368    7643    7647    7713    7717    7745    7849
WB_M[].XOR.ZLIT8[]           5030 #  7054    7058    7070    7082    35455
WB_M[].XZ.MM                 5031 #  15398
WB_NOT(M[].ASR.-P)           5032 #  9710    9799    9843
WB_PL*2                      5033 #  20224
WB_PSL                       5034 #  41560   41641   41662   41691   41696   41701   42682
WB_Q                         5035 #  19302   22730   35607   35655   41775   41785
WB_Q Q_M[]                   5036 #
WB_Q-D                       5037 #  15328   15502   15525   15541
WB_Q-M[]                     5038 #  8260    19404   41915
WB_Q-R[]                     5039 #  19352   42097
WB_Q_M[]                     5040 #  9038
WB_RNUM                      5041 #  5380    7132
WB_R[]                       5042 #  8992    16203   16262   20479   23073   23702   24376   25357   27112   27145   27677
                             28086   28278   29748   33509   33642   33776   34535   34688   36522   39478   40199   40555
                             41460   42411   42530   42802   42952   43091   46598   46632   46665
WB_R[]+0                     5043 #
WB_R[]+1                     5044 #  8995    18685   18688   36517
WB_R[]-0                     5045 #
WB_R[]-CONX(4)               5046 #  23085   23253   23725   23829   24027   24115
WB_R[]-D                     5047 #
WB_R[]-M[]                   5048 #  11015   11019   11169   18320   19309   19518   19588   21049   22972   23811
WB_R[]-Q                     5049 #  19357
WB_R[]-XB PC_PC+4            5050 #  24073   24135
WB_R[]-ZEXT(M[])             5052 #  23766
WB_R[]-ZEXT(XB) PC_PC+1      5053 #  24082   24128
WB_R[].AND.Q                 5055 #  19173   19177
WB_R[].NOTAND.Q              5056 #  35239
WB_R[].OR.(M[].ASR.-P)       5057 #  9825
```

```
WB_R[].RR.16                        5058 #  32085
WB_R[].RR.24                        5059 #  38366   38452
WB_R[].RR.8                         5060 #  34719
WB_R[].RR.SIZ                       5061 #  16486
WB_R[].SR.5(IF MDR IS 0)            5062 #  18400   45056
WB_R[].XOR.Q                        5063 #    798   46423   46448
WB_SEXT(M[])-Q                      5064 #      1
WB_SEXT(M[]).ANDNOT.ZLIT0[]         5065 #      1   33465
WB_SEXT(XB) PC_PC+2                 5066 #      2
WB_SEXT(XB)-R[] PC_PC+1             5067 #
WB_SEXT(XB).XOR.MINUS1 PC_PC+1      5069 #
WB_SEXT(XB).XOR.MINUS1 PC_PC+2      5071 #  5467
WB_XB-R[] PC_PC+4                   5073 #  23984   24047
WB_ZEXT(M[])                        5075 #  29570   29574
WB_ZEXT(M[])-R[]                    5076 #  29554
WB_ZEXT(M[]).AND.R[]                5077 #  24225   24231   24256   24262
WB_ZEXT(XB) PC_PC+1                 5078 #  26302
WB_ZEXT(XB)-(R[].RR.16) PC_PC+2     5079 #  7031
WB_ZEXT(XB)-R[] PC_PC+1             5081 #  23978   23988   24039   24313   24342
WB_ZEXT(XB)-R[] PC_PC+2             5083 #
WB_ZLIT0[]-M[]                      5085 #  13682   30225
WCS DISABLED?                       5246 #  39216   39264   39294
WDR_UNROT(M[].OR.ZLIT24[])          5087 #  40179   40217   40232
WDR_UNROT(R[])                      5088 #  39729   39743   39816
WRITE                               3797 #  5563    5568    5630    8291    16931   19291   19297   19389   21457   21538   21611
                                    21711   21812   32964   32969   46870   47144
WRITE (M[] R[]).RR.4                3798 #  30745
WRITE -M[]                          3799 #  14772
WRITE -Q                            3800 #  8373    21624   21726
WRITE CVTNP(M[])                    3801 #  30708
WRITE CVTPN(M[])                    3802 #  30355   30923   30957   30962   30972   30977
WRITE D+R[]+ALKC                    3803 #
WRITE D.OR.ZLIT28[]                 3804 #
WRITE M[]                           3805 #  6900    6907    6914    7226    8459    8471    8501    8606    8695    8780    8824
                                    8869    14767   19643   19726   20197   20424   21464   21482   21618   21705   23258   23306
                                    23310   28171   28593   28622   28632   28809   28831   30125   30130   30289   30535   30852
                                    30856   30860   30864   32843   32847   32925   32929   33011   33030   33040   34435   34494
                                    36360   36776   36812   37360   37368   38005   38482   38486   38662   42393   42550   42565
                                    42570   46878
WRITE M[]+PSLC                      3806 #  41499
WRITE M[]+Q                         3807 #  23473   41852
WRITE M[]+Q+PSLC                    3808 #  8557
WRITE M[]-PSLC                      3809 #  41538
WRITE M[]-Q                         3810 #  41884
WRITE M[]-Q-PSLC                    3811 #  8731
WRITE M[].AND.ZLIT0[]               3812 #  28617   38570
WRITE M[].ANDNOT.Q                  3813 #  42307
WRITE M[].ANDNOT.R[]                3814 #  37352
WRITE M[].ANDNOT.ZLIT8[]            3815 #  20453
WRITE M[].OR.Q                      3816 #  42241
WRITE M[].OR.R[]                    3817 #
WRITE M[].OR.ZLIT0[]                3818 #  28823   28827   28835   30265   30271   30284
WRITE M[].OR.ZLIT28[]               3819 #
WRITE M[].RR.P                      3820 #  41213
WRITE M[].SL.1                      3821 #
```

```
WRITE M[].XOR.Q                 3822 #
WRITE M[].XZ                    3823 #  38039
WRITE NOTREG                    3824 #  8350    8357    8403    8415    8435    8441    8521    8526    8535    8596    8620
                                8685    8709    8769    8813    8857    8914    8931    8942    8986    9044    9049    9072
                                9319    9328    9336    9340    9514    9518    9522    9526    9687    9721    9725    9730
                                9736    9742    9749    9754    9764    9774    9782    9834    9839    10693   10698   10703
                                10753   10765   10775   10821   10830   11259   11479   11486   11668   11676   11687   11697
                                11704   11776   11789   11897   11908   11914   11929   11934   13167   13246   13772   13797
                                14315   14340   14721   14749   14803   14808   14815   14820   15319   15487   15494   16741
                                16754   16769   16781   16810   16825   16866   16875   16893   16949   16955   18241   18245
                                18278   18291   19449   21036   21041   21361   21550   21805   21900   30000   40989   40999
                                41005   41025   41036   41043   41063   41074   41081   41101   41112   41119   41258   41269
                                41280   41566   41571   41627   41656   41762   41771   41820   41830   41836   41859   41891
                                42210   42221   42620   42630
WRITE Q                         3825 #  5578    5595    8420    9501
WRITE Q.NOT                     3826 #  8362
WRITE Q (Q.SL.1).OR.1           3827 #  47359   47381
WRITE R[]                       3828 #  20229   20239   20248   20257   20265   20274   20283   20292   20300   20309   20318
                                20327   20335   20344   20433   20441   23045   23102   23182   23520   29692   29702   29706
                                30045   33952   37993   37997   38001   42375   42383   42537   42545   46170   46175   47052
                                47059   47331
WRITE R[]+CONX(4)               3829 #  21147
WRITE R[]-D-ALKC                3830 #
WRITE R[]-M[]                   3831 #  21733   21906   21919
WRITE R[]-M[]-1                 3832 #  41191
WRITE XB PC_PC+1                3833 #
WRITE XB PC~PC+4                3835 #  5671
WRITE ZLITO[]                   3837 #  20468   25713   26895   28296   30373   30377   30387   30531   30609   30618   31744
                                38009
WRITE(FLAG1)?                   5247 #  11923
WRITE.LONG                      3838 #  18722   18733
WRITE.LONG D                    3839 #  18711
WRITE.LONG M[].ANDNOT.Q         3840 #  19116   19120
WRITE.LONG M[].OR.Q             3841 #  19108   19112
WRITE.LONG.NOTRAP               3842 #  40216   40231
WRITE.NOTRAP                    3843 #  39673   39683   39817
WRITE.PHY                       3844 #  40178
WRITE.PHY M[]                   3845 #  36875   36885
WRITE.PHY R[]                   3846 #  36857   36899   36911   36929   36935   36941
WRITE.SECOND                    3847 #  39687   39730   39744
WRITE.SECOND.UL                 3848 #  39677
WRITE.UL M[]                    3849 #  22015   22057   22092
WRITE.UL M[]+Q                  3850 #  8652
WRITE.UL M[].ANDNOT.Q           3851 #  19128
WRITE.UL M[].ANDNOT.ZLITO[]     3854 #  26211
WRITE.UL M[].OR.Q               3852 #  19124
WRITE.UL M[].OR.ZLITO[]         3853 #  21997   22002
WX(SIZ).EQ.0?                   5248 #  8927
WX.EQ.0?                        5249 #  5335    5500    5540    5574    5636    5682    5694    5742    5785    5807    5947
                                6007    6229    6281    6285    6299    6313    6327    6341    6355    6369    6383    6397
                                6411    6425    6499    6508    7055    7059    7071    7083    7166    7231    7318    7326
                                7334    7342    7349    7357    7365    7373    7378    7482    7695    7699    7714    7718
                                7746    7850    7924    7974    8068    8076    8080    8199    8920    9698    9703    9711
                                9717    11466   12637   14734   15162   15170   15504   15510   15698   15997   16461   16467
                                18236   19087   19154   19178   19237   19469   19478   19483   20177   20350   20355   20360
```

Cross Reference Listing - Macro Names

	20365	20370	20375	20379	20384	20389	20394	20399	20404	20409	20540	20573
	20608	20618	20627	20637	20647	20657	20667	20676	20686	20696	20706	20716
	20955	21385	21395	21604	21665	21698	21787	21814	21868	21889	21952	21979
	21994	22012	22021	22468	22478	22633	22684	22688	22730	23056	23111	23261
	23531	23536	23702	23767	23812	23880	23985	24048	24074	24136	24216	24220
	24227	24233	24308	24708	24797	24835	25612	26111	26465	27788	27792	28086
	28090	28477	28500	28554	28598	28797	29567	29660	29664	29670	29674	29822
	30249	30499	30648	30691	30696	30700	30735	30746	30836	30848	30904	30942
	30948	30983	30991	31013	31033	31039	31043	31062	31112	32880	33027	33467
	33673	33677	34021	34147	34536	34838	34864	34887	34891	35095	35169	35205
	35214	35240	35456	35532	35536	36031	36069	36479	36496	36563	36580	36723
	36749	37694	37698	37737	37744	39112	39347	39376	41342	41346	41435	41439
	41969	41973	42114	42695	46239	46247	46301	46309	46372	46377	46382	46398
	46423	46448	46472	46483	46529	46534	46539	46566	46573	46580	46587	46593
	46598	46605	46612	46619	46626	46632	46639	46646	46652	46659	46665	46672
	46679	46686	46693	46735	46883	46890	46919	46952	47009	47019	47066	47091
	47098	47109	47162	47173	47208	47219	47248	47259	47299	47353	47375	
WX.NE.0?	5250 #	5468	5716	5761	5824	6464	6471	6478	6647	6861	7314	7322
	7330	7338	7345	7353	7361	7369	7644	7648	7963	8031	8937	9796
	9800	9817	9821	9826	9844	11770	11781	11813	12526	12533	12561	12586
	14701	15072	15088	15923	16516	16991	17238	17482	17530	19083	19174	19242
	20598	20751	21716	21772	21907	21959	22473	23205	23333	23821	24004	24019
	24092	24107	24258	24264	24324	24635	24684	24764	25065	25801	25817	25827
	25837	25885	25889	25896	25907	25920	26303	27087	27118	27122	27126	27130
	27196	27332	27427	27754	27797	28594	28606	28618	28623	28633	28636	28640
	28765	28774	28793	29093	29142	29479	29542	29712	29717	29855	29949	30190
	30781	31456	31588	31650	32629	32635	32639	32645	32655	32671	32687	32702
	32707	32857	32862	32884	32916	32920	32938	32942	32948	32979	32992	32995
	33046	33050	33078	33082	34072	34089	34150	34260	35567	36052	36905	37719
	37831	37845	39275	41355	42083	42479	46183	47574	47721	47732	47758	48028
	48037											
WX<15-0>.NE.0?	5240 #	20216	20797	21311								
WX<31-16>.NE.0?	5241 #	35299										
WX_CONX.SIZ	5089 #	16849										
ZEXT(M[]).EQ.0?	5252 #	9706										

; This page intentionally left blank.

```
ALU.GROUP       3655 #  5381  5492  6135  6145  6695  7132  7955  8020  8036  8041  8046
                8982    8989  9112  9121  9401  9486 12633 14717 14745 14859 14875 15386
                16742   16755 16771 16782 16812 16827 16852 16868 16877 16883 16894 16917
                16933   16939 16950 16957 16980 17102 17254 17299 20156 20801 20950 21315
                21350   21412 21984 22025 22050 22099 23744 23748 23752 23792 23797 23858
                23862   23866 24661 25173 25725 26871 26880 28959 29209 29765 29846 30029
                31169   31309 31740 31748 31865 33245 33559 34704 35051 35055 35387 35543
                35651   35660 35779 35783 35802 35810 35897 35948 36010 36083 36088 36097
                36121   36248 36345 36350 36619 36796 36896 37182 37189 37208 37213 37219
                37336   37690 37709 37804 37827 37841 37864 37918 38035 38084 38098 38149
                38206   38219 38362 38449 38529 38534 38556 38763 38817 38894 38898 38906
                38924   38962 39034 39154 39944 39955 40858 41362 41385 41561 41643 41664
                41693   41698 41703 42041 42171 42508 42670 42683 47278 47570
CHAR.R1H        2415 #  22431 22504 22563 22618 22668 22734 24738 25018 25199 25279 25360
CHAR.R1L        2414 #  22431 22504 22563 22618 22668 22734 24738 25018 25199 25279 25360
CHAR.R2H        2417 #  22431 22504 22563 22618 22668 22734 24738 25018 25199 25279 25360
CHAR.R2L        2416 #  22431 22504 22563 22618 22668 22734 24738 25018 25199 25279 25360
CHAR.R3H        2419 #  22431 22504 22563 22618 22668 22734 24738 25018 25199 25279 25360
CHAR.R3L        2418 #  22431 22504 22563 22618 22668 22734 24738 25018 25199 25279 25360
CMODE.R1H       2471 #  41208 41223 41249 41299 41402 41483 41502 41522 41541 41563 41589
                41621   41645 41666 41685 41744 41757 41865 41897 41928 41950 42079 42254
                42287   42320 42349 42371 42430 42481 42505 42533 42561 42600 42647 42811
                42961   43100
CMODE.R1L       2470 #  41208 41223 41249 41299 41402 41483 41502 41522 41541 41563 41589
                41621   41645 41666 41685 41744 41757 41865 41897 41928 41950 42079 42254
                42287   42320 42349 42371 42430 42481 42505 42533 42561 42600 42647 42811
                42961   43100
CMODE.R2H       2473 #  41208 41223 41249 41299 41402 41483 41502 41522 41541 41563 41589
                41621   41645 41666 41685 41744 41757 41865 41897 41928 41950 42079 42254
                42287   42320 42349 42371 42430 42481 42505 42533 42561 42600 42647 42811
                42961   43100
CMODE.R2L       2472 #  41208 41223 41249 41299 41402 41483 41502 41522 41541 41563 41589
                41621   41645 41666 41685 41744 41757 41865 41897 41928 41950 42079 42254
                42287   42320 42349 42371 42430 42481 42505 42533 42561 42600 42647 42811
                42961   43100
CMODE.R3H       2475 #  41208 41223 41249 41299 41402 41483 41502 41522 41541 41563 41589
                41621   41645 41666 41685 41744 41757 41865 41897 41928 41950 42079 42254
                42287   42320 42349 42371 42430 42481 42505 42533 42561 42600 42647 42811
                42961   43100
CMODE.R3L       2474 #  41208 41223 41249 41299 41402 41483 41502 41522 41541 41563 41589
                41621   41645 41666 41685 41744 41757 41865 41897 41928 41950 42079 42254
                42287   42320 42349 42371 42430 42481 42505 42533 42561 42600 42647 42811
                42961   43100
CONSOL.R1H      2354 #  6124
CONSOL.R1L      2353 #  6124
CONSOL.R2H      2356 #
CONSOL.R2L      2355 #
CONSOL.R3H      2358 #
CONSOL.R3L      2357 #
CONTRL.R1H      2386 #  18905 18957 19006 19072 19156 19244 19286 19345 19384 19446 19521
                19555   19636 19695 19717 19735
CONTRL.R1L      2385 #  18905 18957 19006 19072 19156 19244 19286 19345 19384 19446 19521
                19555   19636 19695 19717 19735
CONTRL.R2H      2388 #  18905 18957 19006 19072 19156 19244 19286 19345 19384 19446 19521
                19555   19636 19695 19717 19735
```

Cross Reference Listing - Expression Names

CONTRL.R2L
2387 # 18905 18957 19006 19072 19156 19244 19286 19345 19384 19446 19521
19555 19636 19695 19717 19735

CONTRL.R3H
2390 # 18905 18957 19006 19072 19156 19244 19286 19345 19384 19446 19521
19555 19636 19695 19717 19735

CONTRL.R3L
2389 # 18905 18957 19006 19072 19156 19244 19286 19345 19384 19446 19521
19555 19636 19695 19717 19735

DECMAL.R1H
2423 # 25614 25787 26107 26448 27004 27661 29077 30007 30027 30121 30217
30344 30445 31216 31438 32032 33105

DECMAL.R1L
2422 # 25614 25787 26107 26448 27004 27661 29077 30007 30027 30121 30217
30344 30445 31216 31438 32032 33105

DECMAL.R2H
2425 # 25614 25787 26107 26448 27004 27661 29077 30007 30027 30121 30217
30344 30445 31216 31438 32032 33105

DECMAL.R2L
2424 # 25614 25787 26107 26448 27004 27661 29077 30007 30027 30121 30217
30344 30445 31216 31438 32032 33105

DECMAL.R3H
2427 # 29465 29816

DECMAL.R3L
2426 # 29465 29816

DQ.0
3659 # 5381 5381 5492 5492 6135 6135 6145 6145 6695 6695 7132
7132 7955 7955 8020 8020 8036 8036 8041 8041 8046 8046 8982
8982 8989 8989 9112 9112 9121 9121 9401 9401 9486 9486 12633
12633 14717 14717 14745 14745 14859 14859 14875 14875 15386 15386 16742
16742 16755 16755 16771 16771 16782 16782 16812 16812 16827 16827 16852
16852 16868 16868 16877 16877 16883 16883 16894 16894 16917 16917 16933
16933 16939 16939 16950 1695(16957 16957 16980 16980 17102 17102 17254
17254 17299 17299 20156 20156 20801 20801 20950 20950 21315 21315 21350
21350 21412 21412 21984 21984 22025 22025 22050 22050 22099 22099 23744
23744 23748 23748 23752 23752 23792 23792 23797 23797 23858 23858 23862
23862 23866 23866 24661 24661 25173 25173 25725 25725 26871 26871 26880
26880 28959 28959 29209 29209 29765 29765 29846 29846 30029 30029 31169
31169 31309 31309 31740 31740 31748 31748 31865 31865 33245 33245 33559
33559 34704 34704 35051 35051 35055 35055 35387 35387 35543 35543 35651
35651 35660 35660 35779 35779 35783 35783 35802 35802 35810 35810 35897
35897 35948 35948 36010 36010 36083 36083 36088 36088 36097 36097 36121
36121 36248 36248 36345 36345 36350 36350 36619 36619 36796 36796 36896
36896 37182 37182 37189 37189 37208 37208 37213 37213 37219 37219 37336
37336 37690 37690 37709 37709 37804 37804 37827 37827 37841 37841 37864
37864 37918 37918 38035 38035 38084 38084 38098 38098 38149 38149 38206
38206 38219 38219 38362 38362 38449 38449 38529 38529 38534 38534 38556
38556 38763 38763 38817 38817 38894 38894 38898 38898 38906 38906 38924
38924 38962 38962 39034 39034 39154 39154 39944 39944 39955 39955 40858
40858 41362 41362 41385 41385 41561 41561 41643 41643 41664 41664 41693
41693 41698 41698 41703 41703 42041 42041 42171 4217[illegible] 42508 42508 42670
42670 42683 42683 47278 47278 47570 47570

DQ.1
3660 # 5381 5381 5492 5492 6135 6135 6145 6145 6695 6695 7132
7132 7955 7955 8020 8020 8036 8036 8041 8041 8046 8046 8982
8982 8989 8989 9112 9112 9121 9121 9401 9401 9486 9486 12633
12633 14717 14717 14745 14745 14859 14859 14875 14875 15386 15386 16742
16742 16755 16755 16771 16771 16782 16782 16812 16812 16827 16827 16852
16852 16868 16868 16877 16877 16883 16883 16894 16894 16917 16917 16933
16933 16939 16939 16950 16950 16957 16957 16980 16980 17102 17102 17254
17254 17299 17299 20156 20156 20801 20801 20950 20950 21315 21315 21350
21350 21412 21412 21984 21984 22025 22025 22050 22050 22099 22099 23744
23744 23748 23748 23752 23752 23792 23792 23797 23797 23858 23858 23862
23862 23866 23866 24661 24661 25173 25173 25725 25725 26871 26871 26880
26880 28959 28959 29209 29209 29765 29765 29846 29846 30029 30029 31169
31169 31309 31309 31740 31740 31748 31748 31865 31865 33245 33245 33559

```
                33559   34704   34704   35051   35051   35055   35055   35387   35387   35543   35543   35651
                35651   35660   35660   35779   35779   35783   35783   35802   35802   35810   35810   35897
                35897   35948   35948   36010   36010   36083   36083   36088   36088   36097   36097   36121
                36121   36248   36248   36345   36345   36350   36350   36619   36619   36796   36796   36896
                36896   37182   37182   37189   37189   37208   37208   37213   37213   37219   37219   37336
                37336   37690   37690   37709   37709   37804   37804   37827   37827   37841   37841   37864
                37864   37918   37918   38035   38035   38084   38084   38098   38098   38149   38149   38206
                38206   38219   38219   38362   38362   38449   38449   38529   38529   38534   38534   38556
                38556   38763   38763   38817   38817   38804   38894   38898   38898   38906   38906   38924
                38924   38962   38962   39034   39034   39154   39154   39944   39944   39955   39955   40858
                40858   41362   41362   41385   41385   41561   41561   41643   41643   41664   41664   41693
                41693   41698   41698   41703   41703   42041   42041   42171   42171   42508   42508   42670
                42670   42683   42683   47278   47278   47570   47570
EDIT.R1H        2431 #  33468   33494   34656
EDIT.R1L        2430 #  33468   33494   34656
EDIT.R2H        2433 #  33468   33494   34656
EDIT.R2L        2432 #  33468   33494   34656
EDIT.R3H        2435 #  33468   33494   34656
EDIT.R3L        2434 #  33468   33494   34656
FLOAT.R1H       2370 #  10686   10756   10810   10917   10984   11095   11124   11228   11342   11354   11450
                11653   11758   11876   12048   12872   13066   13304   13475   13566   13802   13889   14438
                14680   14995   15324   15473   15701   16207   16227   16416   16437   17468   17604
FLOAT.R1L       2369 #  10686   10756   10810   10917   10984   11095   11124   11228   11342   11354   11450
                11653   11758   11876   12048   12872   13066   13304   13475   13566   13802   13889   14438
                14680   14995   15324   15473   15701   16207   16227   16416   16437   17468   17604
FLOAT.R2H       2372 #  10686   10756   10810   10917   10984   11095   11124   11228   11342   11354   11450
                11653   11758   11876   12048   12872   13066   13304   13475   13566   13802   13889   14438
                14680   14995   15324   15473   15701   16207   16227   16416   16437   17468   17604
FLOAT.R2L       2371 #  10686   10756   10810   10917   10984   11095   11124   11228   11342   11354   11450
                11653   11758   11876   12048   12872   13066   13304   13475   13566   13802   13889   14438
                14680   14995   15324   15473   15701   16207   16227   16416   16437   17468   17604
FLOAT.R3H       2374 #  10686   10756   10810   10917   10984   11095   11124   11228   11342   11354   11450
                11653   11758   11876   12048   12872   13066   13304   13475   13566   13802   13889   14438
                14680   14995   15324   15473   15701   16207   16227   16416   16437   17468   17604
FLOAT.R3L       2373 #  10686   10756   10810   10917   10984   11095   11124   11228   11342   11354   11450
                11653   11758   11876   12048   12872   13066   13304   13475   13566   13802   13889   14438
                14680   14995   15324   15473   15701   16207   16227   16416   16437   17468   17604
GHFLT.R1H       2509 #  47461   47824
GHFLT.R1L       2508 #  47461   47824
GHFLT.R2H       2511 #  47461   47824
GHFLT.R2L       2510 #  47461   47824
GHFLT.R3H       2513 #  47461   47824
GHFLT.R3L       2512 #  47461   47824
IANDE.R1H       2455 #  37176   38633   38708   39331
IANDE.R1L       2454 #  37176   38633   38708   39331
IANDE.R2H       2457 #  37176   38633   38708   39331
IANDE.R2L       2456 #  37176   38633   38708   39331
IANDE.R3H       2459 #  37176   38633   38708   39331
IANDE.R3L       2458 #  37176   38633   38706   39331
INIT            2534 #  5312 #  5547    5564    5569    5579    5596    5622    5630    5672    5687    5733
                5754    5791    5800    5818    5826 #  6779    6787    6796    6900    6907    6914    7227
                8292    8351    8357    8363    8373    8405    8416    8420    8436    8441    8460    8472
                8501    8522    8528    8536    8558    8596    8606    8620    8645    8653    8685    8695
                8709    8732    8740    8780    8814    8824    8858    8869    8916    8932    8942    8986
                9045    9050    9072    9319    9329    9336    9341    9502    9515    9519    9523    9527
```

Cross Reference Listing - Expression Names

9687	9721	9727	9731	9737	9743	9750	9756	9765	9775	9782	9835
9840	10694	10699	10704	10754	10767	10777	10822	10831	11262	11482	11489
11672	11678	11691	11699	11706	11777	11790	11899	11910	11917	11930	11935
13169	13248	13774	13800	14317	14343	14723	14751	14769	14774	14804	14809
14816	14822	15322	15489	15496	15719	15760	16021	16035	16219	16611	16618
16643	16742	16755	16771	16782	16812	16827	16868	16877	16894	16933	16950
16957	17010	17014	17020	17041	17081	17091	17261	17266	17286	17557	18241
18245	18279	18291	18410	18414	18418	18427	18439	18451	18695	18711	18715
18723	18734	19042	19053	19077	19109	19113	19117	19121	19125	19129	19293
19299	19391	19451	19540	19546	19645	19728	19739	20112	20118	20197	20230
20240	20249	20258	20266	20275	20284	20293	20301	20310	20319	20328	20336
20345	20425	20434	20441	20455	20469	20536	20581	20586	20591	20614	20623
20633	20643	20653	20663	20672	20682	20692	20702	20712	20721	20761	21037
21041	21147	21272	21362	21446	21458	21464	21483	21527	21532	21539	21550
21612	21618	21625	21659	21705	21712	21726	21734	21780	21805	21814	21861
21878	21900	21907	21920	21965	21999	22004	22017	22050	22058	22093	23046
23103	23183	23258	23307	23311	23475	23521	23757	23804	24202	24206	24481
25714	26212	26345	26896	28172	28298	28594	28619	28623	28633	28810	28824
28828	28832	28836	29398	29694	29703	29707	29843	29912	30002	30046	30118
30127	30132	30215	30267	30273	30286	30290	30356	30374	30380	30388	30451
30532	30536	30611	30620	30710	30747	30853	30857	30861	30865	30924	30959
30963	30974	30978	31074	31745	32408	32412	32435	32456	32475	32491	32844
32848	32926	32930	32965	32970	33011	33031	33041	33500	33515	33953	34380
34437	34495	36139	36324	36361	36688	36697	36712	36719	36736	36739	36759
36763	36767	36777	36788	36813	36818	36858	36871	36875	36882	36885	36900
36911	36929	36935	36942	37213	37219	37353	37361	37369	37994	37998	38002
38006	38011	38040	38286	38483	38487	38572	38664	39335	39338	39649	39653
39674	39678	39684	39688	39731	39745	39817	40180	40218	40232	40313	40518
40581	40600	40633	40991	41001	41007	41027	41038	41045	41065	41076	41083
41103	41114	41121	41193	41215	41260	41271	41282	41500	41539	41568	41573
41628	41657	41763	41772	41821	41831	41837	41854	41860	41886	41892	42211
42222	42243	42309	42376	42384	42393	42538	42546	42551	42566	42571	42604
42621	42631	42652	42716	42821	42831	42847	42967	42977	42993	43169	43216
43243	43252	44416	44426	44438	44444	44456	44539	44545	44635	44641	44724
44730	44884	44891	44898	44904	44915	45027	45040	45131	45138	45240	45247
45338	45420	45511	45525	45542	45548	45554	45565	45654	45664	45767	45775
45781	45790	45802	45881	45974	46071	46079	46085	46171	46176	46768	46871
46879	47053	47059	47144	47150	47197	47237	47332	47346	47361	47366	47382

INIT.R1H	2346 #	5311									
INIT.R1L	2345 #	5311									
INIT.R2H	2348 #	5311									
INIT.R2L	2347 #	5311									
INIT.R3H	2350 #	5311									
INIT.R3L	2349 #	5311									
INTLOG.R1H	2362 #	8294	8309	8407	8457	8469	8494	8508	8530	8560	8642	8734
	8922	9029	9312	9390	9672							
INTLOG.R1L	2361 #	8294	8309	8407	8457	8469	8494	8508	8530	8560	8642	8734
	8922	9029	9312	9390	9672							
INTLOG.R2H	2364 #	8294	8309	8407	8457	8469	8494	8508	8530	8560	8642	8734
	8922	9029	9312	9390	9672							
INTLOG.R2L	2363 #	8294	8309	8407	8457	8469	8494	8508	8530	8560	8642	8734
	8922	9029	9312	9390	9672							
INTLOG.R3H	2366 #	8294	8309	8407	8457	8469	8494	8508	8530	8560	8642	8734
	8922	9029	9312	9390	9672							
INTLOG.R3L	2365 #	8294	8309	8407	8457	8469	8494	8508	8530	8560	8642	8734

Cross Reference Listing - Expression Names

	8922	9029	9312	9390	9672							
IRD1.R1H	2495 #	8968	9090	16194	16221	16718	16757	16829	16854	16896	16959	16982
	17066	17191	17228	17597	18862	18922	18965	19602	19653	19729	20526	20896
	20925	20946	24731	25011	25192	25272	25353	29070	30021	31210	32025	34650
	36663	36837	38701	39326	46900	47075						
IRD1.R1L	2494 #	8968	9090	16194	16221	16718	16757	16829	16854	16896	16959	16982
	17066	17191	17228	17597	18862	18922	18965	19602	19653	19729	20526	20896
	20925	20946	24731	25011	25192	25272	25353	29070	30021	31210	32025	34650
	36663	36837	38701	39326	46900	47075						
IRDX.R1H	2499 #	8236	8257	8275	8288	8303	8329	8346	8400	8432	8451	8464
	8488	8503	8518	8549	8586	8616	8632	8675	8705	8723	8759	8803
	8847	8894	9014	9266	9331	9371	9662	10676	10728	10801	10906	10972
	11089	11118	11219	11246	11315	11348	11436	11546	11625	11753	11973	12030
	12826	13045	13143	13219	13444	13757	13776	13857	14300	14319	14414	14630
	14931	15295	15468	15675	16411	16432	17438	18192	18211	18268	18309	18647
	19018	19038	19138	19233	19271	19326	19377	19440	19507	19548	19712	20094
	21001	21096	21198	21346	21380	21431	21509	21580	21646	21755	21845	22413
	22492	22552	22592	22648	22693	25597	26085	26413	29453	29805	30079	30171
	30321	30424	31406	33462	33477	34948	35738	36224	36461	36545	38605	40982
	41017	41055	41093	41131	41166	41203	41217	41243	41273	41294	41397	41476
	41496	41515	41535	41557	41583	41616	41638	41659	41674	41738	41752	41797
	41809	41849	41881	41912	41944	42073	42184	42238	42271	42304	42331	42365
	42422	42456	42473	42499	42525	42553	42593	42638	42731	42877	43018	
IRDX.R1L	2498 #	8236	8257	8275	8288	8303	8329	8346	8400	8432	8451	8464
	8488	8503	8518	8549	8586	8616	8632	8675	8705	8723	8759	8803
	8847	8894	9014	9266	9331	9371	9662	10676	10728	10801	10906	10972
	11089	11118	11219	11246	11315	11348	11436	11546	11625	11753	11973	12030
	12826	13045	13143	13219	13444	13757	13776	13857	14300	14319	14414	14630
	14931	15295	15468	15675	16411	16432	17438	18192	18211	18268	18309	18647
	19018	19038	19138	19233	19271	19326	19377	19440	19507	19548	19712	20094
	21001	21096	21198	21346	21380	21431	21509	21580	21646	21755	21845	22413
	22492	22552	22592	22648	22693	25597	26085	26413	29453	29805	30079	30171
	30321	30424	31406	33462	33477	34948	35738	36224	36461	36545	38605	40982
	41017	41055	41093	41131	41166	41203	41217	41243	41273	41294	41397	41476
	41496	41515	41535	41557	41583	41616	41638	41659	41674	41738	41752	41797
	41809	41849	41881	41912	41944	42073	42184	42238	42271	42304	42331	42365
	42422	42456	42473	42499	42525	42553	42593	42638	42731	42877	43018	
IRDX.R2H	2501 #	8236	8257	8275	8288	8303	8329	8346	8400	8432	8451	8464
	8488	8503	8518	8549	8586	8616	8632	8675	8705	8723	8759	8803
	8847	8894	9014	9266	9331	9371	9662	10676	10728	10801	10906	10972
	11089	11118	11219	11246	11315	11348	11436	11546	11625	11753	11973	12030
	12826	13045	13143	13219	13444	13757	13776	13857	14300	14319	14414	14630
	14931	15295	15468	15675	16411	16432	17438	18192	18211	18268	18309	18647
	19018	19038	19138	19233	19271	19326	19377	19440	19507	19548	19712	20094
	21001	21096	21198	21346	21380	21431	21509	21580	21646	21755	21845	22413
	22492	22552	22592	22648	22693	25597	26085	26413	29453	29805	30079	30171
	30321	30424	31406	33462	33477	34948	35738	36224	36461	36545	38605	40982
	41017	41055	41093	41131	41166	41203	41217	41243	41273	41294	41397	41476
	41496	41515	41535	41557	41583	41616	41638	41659	41674	41738	41752	41797
	41809	41849	41881	41912	41944	42073	42184	42238	42271	42304	42331	42365
	42422	42456	42473	42499	42525	42553	42593	42638	42731	42877	43018	
IRDX.R2L	2500 #	8236	8257	8275	8288	8303	8329	8346	8400	8432	8451	8464
	8488	8503	8518	8549	8586	8616	8632	8675	8705	8723	8759	8803
	8847	8894	9014	9266	9331	9371	9662	10676	10728	10801	10906	10972
	11089	11118	11219	11246	11315	11348	11436	11546	11625	11753	11973	12030

Cross Reference Listing - Expression Names

```
                12826   13045   13143   13219   13444   13757   13776   13857   14300   14319   14414   14630
                14931   15295   15468   15675   16411   16432   17438   18192   18211   18268   18309   18647
                19018   19038   19138   19233   19271   19326   19377   19440   19507   19548   19712   20094
                21001   21096   21198   21346   21380   21431   21509   21580   21646   21755   21845   22413
                22492   22552   22592   22648   22693   25597   26085   26413   29453   29805   30079   30171
                30321   30424   31406   33462   33477   34948   35738   36224   36461   36545   38605   40982
                41017   41055   41093   41131   41166   41203   41217   41243   41273   41294   41397   41476
                41496   41515   41535   41557   41583   41616   41638   41659   41674   41738   41752   41797
                41809   41849   41881   41912   41944   42073   42184   42238   42271   42304   42331   42365
                42422   42456   42473   42499   42525   42553   42593   42638   42731   42877   43018
MICROREV        5 #     36151
MISQUE.R1H      2407 #  20932   20952   21028   21100   21203   21352   21402   21435   21523   21590   21656
                21769   21859
MISQUE.R1L      2406 #  20932   20952   21028   21100   21203   21352   21402   21435   21523   21590   21656
                21769   21859
MISQUE.R2H      2409 #  20932   20952   21028   21100   21203   21352   21402   21435   21523   21590   21656
                21769   21859
MISQUE.R2L      2408 #  20932   20952   21028   21100   21203   21352   21402   21435   21523   21590   21656
                21769   21859
MISQUE.R3H      2411 #  20932   20952   21028   21100   21203   21352   21402   21435   21523   21590   21656
                21769   21859
MISQUE.R3L      2410 #  20932   20952   21028   21100   21203   21352   21402   21435   21523   21590   21656
                21769   21859
MM.R1H          2463 #  39640
MM.R1L          2462 #  39640
MM.R2H          2465 #  39640
MM.R2L          2464 #  39640
MM.R3H          2467 #  39640
MM.R3L          2466 #  39640
MPRCHM.R1H      2439 #  34833   34970   35766   36244
MPRCHM.R1L      2438 #  34833   34970   35766   36244
MPRCHM.R2H      2441 #  34833   34970   35766   36244
MPRCHM.R2L      2440 #  34833   34970   35766   36244
MPRCHM.R3H      2443 #  34833   34970   35766   36244
MPRCHM.R3L      2442 #  34833   34970   35766   36244
MUX.09          3657 #  5381    5492    6135    6145    6695    7132    7955    8020    8036    8041    8046
                8982    8989    9112    9121    9401    9486    12633   14717   14745   14859   14875   15386
                16742   16755   16771   16782   16812   16827   16852   16868   16877   16883   16894   16917
                16933   16939   16950   16957   16980   17102   17254   17299   20156   20801   20950   21315
                21350   21412   21984   22025   22050   22099   23744   23748   23752   23792   23797   23858
                23862   23866   24661   25173   25725   26871   26880   28959   29209   29765   29846   30029
                31169   31309   31740   31748   31865   33245   33559   34704   35051   35055   35387   35543
                35651   35660   35779   35783   35802   35810   35897   35948   36010   36083   36088   36097
                36121   36248   36345   36350   36619   36796   36896   37182   37189   37208   37213   37219
                37336   37690   37709   37804   37827   37841   37864   37918   38035   38084   38098   38149
                38206   38219   38362   38449   38529   38534   38556   38763   38817   38894   38898   38906
                38924   38962   39034   39154   39944   39955   40858   41362   41365   41561   41643   41664
                41693   41698   41703   42041   42171   42508   42670   42683   47278   47570
MUX.0D          3658 #  5381    5492    6135    6145    6695    7132    7955    8020    8036    8041    8046
                8982    8989    9112    9121    9401    9486    12633   14717   14745   14859   14875   15386
                16742   16755   16771   16782   16812   16827   16852   16868   16877   16883   16894   16917
                16933   16939   16950   16957   16980   17102   17254   17299   20156   20801   20950   21315
                21350   21412   21984   22025   22050   22099   23744   23748   23752   23792   23797   23858
                23862   23866   24661   25173   25725   26871   26880   28959   29209   29765   29846   30029
                31169   31309   31740   31748   31865   33245   33559   34704   35051   35055   35387   35543
```

Cross Reference Listing - Expression Names

```
                    35651   35660   35779   35783   35802   35810   35897   35948   36010   36083   36088   36097
                    36121   36248   36345   36350   36619   36796   36896   37182   37189   37208   37213   37219
                    37336   37690   37709   37804   37827   37841   37864   37918   38035   38084   38098   38149
                    38206   38219   38362   38449   38529   38534   38556   38763   38817   38894   38898   38906
                    38924   38962   39034   39154   39944   39955   40858   41362   41385   41561   41643   41664
                    41693   41698   41703   42041   42171   42508   42670   42683   47273   47570
MV.R1H              2479 #  46141   46905   47078
MV.R1L              2478 #  46141   46905   47078
MV.R2H              2481 #  46141   46905   47078
MV.R2L              2480 #  46141   46905   47078
MV.R3H              2483 #  46141   46905   47078
MV.R3L              2482 #  46141   46905   47078
OSR.R1H             2487 #  44344
OSR.R1L             2486 #  44344
OSR.R2H             2489 #  44344
OSR.R2L             2488 #  44344
OSR.R3H             2491 #  44344
OSR.R3L             2490 #  44344
OTHERS              3669 #   5381    5492    6135    6145    6695    7132    7955    8020    8036    8041    8046
                     8982    8989    9112    9121    9401    9486   12633   14717   14745   14859   14875   15386
                    16742   16755   16771   16782   16812   16827   16852   16868   16877   16883   16894   16917
                    16933   16939   16950   16957   16980   17102   17254   17299   20156   20801   20950   21315
                    21350   21412   21984   22025   22050   22099   23744   23748   23752   23792   23797   23858
                    23862   23866   24661   25173   25725   26871   26880   28959   29209   29765   29846   30029
                    31169   31309   31740   31748   31865   33245   33559   34704   35051   35055   35387   35543
                    35651   35660   35779   35783   35802   35810   35897   35948   36010   36083   36088   36097
                    36121   36248   36345   36350   36619   36796   36896   37182   37189   37208   37213   37219
                    37336   37690   37709   37804   37827   37841   37864   37918   38035   38084   38098   38149
                    38206   38219   38362   38449   38529   38534   38556   38763   38817   38894   38898   38906
                    38924   38962   39034   39154   39944   39955   40858   41362   41385   41561   41643   41664
                    41693   41698   41703   42041   42171   42508   42670   42683   47278   47570
PCALL.R1H           2399 #  20101   20533
PCALL.R1L           2398 #  20101   20533
PCALL.R2H           2401 #  20101   20533
PCALL.R2L           2400 #  20101   20533
PCALL.R3H           2403 #  20101   20533
PCALL.R3L           2402 #  20101   20533
PRLDSV.R1H          2447 #  36471   36555   36670   36844
PRLDSV.R1L          2446 #  36471   36555   36670   36844
PRLDSV.R2H          2449 #  36471   36555   36670   36844
PRLDSV.R2L          2448 #  36471   36555   36670   36844
PRLDSV.R3H          2451 #  36471   36555   36670   36844
PRLDSV.R3L          2450 #  36471   36555   36670   36844
V00.30              3558 #   5519    5526    5547    5553    5611    5636    5659    5682    5699    5712    5726
                     5733    5750    5762    5791    5796    5800    5888    5907    5920    5927    5934    6142
                     6236    6686    6746    6872    6876    6974    7014    7184    7220    8128    8132    8136
                     8139    8411    8455    8467    8532    9055    9063    9498    9511    9760    9770    9778
                     9788    9805   11923   12580   12642   14763   14791   14795   14799   15551   15714   15719
                    15760   15992   16021   16035   16141   16219   16284   16534   16611   16618   16643   16776
                    16888   16927   16943   17010   17014   17020   17041   17076   17081   17091   17160   17169
                    17173   17177   17261   17266   17286   17552   18406   18410   18414   18418   18692   18703
                    18718   18726   19065   19150   19529   19640   19723   19733   20099   20124   20135   20192
                    20195   20203   20445   20504   20509   20514   20531   20536   20545   20570   20581   20586
                    20591   20614   20623   20633   20643   20653   20663   20672   20682   20692   20702   20712
                    20721   20735   20739   20746   20761   20816   20820   20901   20930   21102   21112   21154
```

Cross Reference Listing - Expression Names

21160	21206	21216	21234	21255	21272	21434	21439	21443	21450	21454	21461
21468	21472	21513	21521	21527	21532	21536	21543	21546	21585	21589	21600
21608	21615	21621	21651	21655	21662	21669	21678	21682	21693	21702	21708
21723	21730	21758	21764	21768	21791	21794	21801	21808	21848	21854	21858
21871	21874	21881	21893	21896	21903	21916	21989	22046	22447	22773	22991
23061	23065	23116	23120	23198	23201	23248	23266	23270	23326	23329	23541
23545	23558	23563	23826	23835	23839	23885	23889	24196	24572	24810	24821
24831	24855	25114	25128	25322	25364	25661	25692	25718	25807	25869	25873
26338	26341	26593	26645	26671	26675	26679	26683	26876	26900	27048	27344
27705	28110	28121	28155	28160	28208	28334	28338	28661	28665	29163	29214
29276	29356	29366	29398	29579	29615	29645	29680	29684	29698	29723	29727
29831	29843	29912	30039	30118	30215	30360	30365	30447	30632	30636	30714
30756	30760	30875	30879	30888	30909	30926	30931	30968	31134	31337	31349
31527	31719	31753	32375	32385	32396	32400	32404	32482	32511	32815	32823
32829	32833	32837	33019	33496	33503	33945	34369	34432	34447	34451	34489
34510	35027	35107	35111	35123	35127	35132	35136	35257	35276	35309	35498
35503	35512	35526	35590	35629	35850	35853	35857	35861	35873	35885	35919
35923	35927	35931	35935	35939	36136	36293	36304	36316	36342	36357	36486
36570	36643	36667	36684	36701	36707	36712	36719	36729	36736	36745	36759
36763	36782	36785	36809	36818	36853	36861	36867	36871	36878	36882	36888
36926	36932	36938	37110	37115	37121	37131	37137	37331	37357	37365	37375
37400	37483	37492	37563	37663	37679	37686	37765	37770	37773	37801	37839
37853	37878	37914	37962	37965	37969	37973	37977	37981	37990	38092	38113
38169	38214	38229	38279	38493	38563	38566	38579	38655	38659	38722	38772
38776	38918	39087	39167	39181	39185	39191	39329	39335	39398	39649	39693
39726	39740	39760	39806	39813	39822	39828	39834	39885	39925	39932	39949
39960	40039	40050	40061	40066	40071	40096	40185	40208	40262	40300	40303
40309	40321	40327	40330	40368	40380	40394	40439	40506	40509	40515	40532
40538	40577	40594	40630	40636	40715	40723	40775	40782	42404	42559	42581
42598	42612	42707	42720	42745	42749	42754	42759	42764	42769	42779	42785
42789	42794	42866	42893	42897	42902	42907	42913	42918	42928	42935	42939
42944	42959	43007	43034	43038	43043	43048	43054	43059	43076	43084	43098
43140	43148	43199	43209	43227	43232	43237	44354	44369	44379	44384	44388
44398	44403	44449	44508	44512	44523	44534	44568	44593	44597	44604	44608
44619	44630	44694	44698	44709	44714	44782	44786	44805	44826	44837	44847
44852	44857	44866	44871	44909	44974	44979	45103	45108	45174	45187	45201
45207	45212	45222	45227	45252	45281	45292	45302	45306	45310	45320	45325
45361	45375	45385	45390	45394	45404	45409	45448	45462	45472	45477	45481
45491	45496	45511	45525	45542	45559	45595	45608	45618	45623	45627	45637
45642	45654	45708	45718	45728	45733	45737	45747	45752	45757	45775	45786
45795	45823	45834	45844	45849	45854	45863	45868	45905	45922	45934	45939
45944	45956	45961	45974	46014	46025	46035	46040	46044	46054	46059	46071
46079	46148	46166	46179	46768	46815	46832	46875	46883	46890	46908	46919
46952	46990	47000	47009	47015	47019	47056	47091	47098	47109	47139	47158
47162	47169	47173	47181	47187	47201	47208	47215	47219	47227	47241	47248
47255	47259	47265	47271	47299	47326	47338	47341	47371	47402	47415	47810
V00.30.41	3560 #	5358	5422	5455	5459	5463	5667	5676	5882	6122	6139	6447
	6657	6702	6969	7027	7042	7045	7190	15340	16294	17470	17488	17578
	17616	18915	19022	19098	19189	19250	19322	19410	19414	19418	19461	19469
	19474	19493	19566	19583	19719	20458	20594	20914	20962	22458	22627	22643
	22674	22722	22789	22994	23050	23053	23107	23187	23450	23454	23527	23551
	23761	23808	23818	23877	23928	24015	24103	24251	24318	24328	24372	24427
	24569	24712	24789	24844	24860	24873	25123	25376	25388	26189	26198	26306
	26333	27724	27728	28849	28866	29228	29238	29339	29370	30025	31126	31245
	31286	31291	31301	31305	32037	33556	33575	33956	34557	34616	34673	35593

Cross Reference Listing - Expression Names

```
                    35617   36527   36715   36841   37256   37346   37380   37388   37755   37776   38106   38807
                    39210   39269   39434   39644   39755   40256   40390   40996   41032   41070   41108   41181
                    41265   41490   41529   41614   41728   41782   41807   41826   41869   41901   42217   42259
                    42324   42357   42398   42435   42439   42443   42465   42493   42626   42655   42855   43001
                    43080   43192   44751   45002   46243   46251   46305   46313   46391   46407   46416   46432
                    46441   46457   46475   46492   46569   46576   46583   46590   46595   46601   46608   46615
                    46622   46629   46635   46642   46648   46655   46662   46668   46675   46682   46689   46696
                    46744   46886   46893   46895   46911   46922   46926   46947   46993   47003   47012   47022
                    47066   47094   47101   47112   47154   47165   47176   47197   47211   47222   47237   47251
                    47262   47295   47302   47356   47378
V00.30.80           3559 #  35783
V00.71              3561 #
V00.71.80           3562 #  6695    12633   15386   20156   20950   21350   21412   30029   33559   34704   35651
                    35660   35779   36248   36345   36350   36619   36796   36896   37182   37189   37336   40858
                    41362   41385   41561   41643   41664   41693   41698   41703   42041   42171   42508   42670
                    42683   47570
V000                3587 #  5358    5422    5455    5459    5463    5519    5526    5547    5553    5611    5636
                    5659    5667    5676    5682    5699    5712    5726    5733    5750    5762    5791    5796
                    5800    5882    5888    5907    5920    5927    5934    6122    6139    6142    6236    6437
                    6657    6686    6695    6702    6746    6872    6876    6969    6974    7014    7027    7042
                    7045    7184    7190    7220    8128    8132    8136    8139    8411    8455    8467    8532
                    9055    9063    9498    9511    9760    9770    9778    9788    9805    11923   12580   12633
                    12642   14763   14791   14795   14799   15340   15386   15551   15714   15719   15760   15992
                    16021   16035   16141   16219   16284   16294   16534   16611   16618   16643   16776   16888
                    16927   16943   17010   17014   17020   17041   17076   17081   17091   17160   17169   17173
                    17177   17261   17266   17286   17470   17488   17552   17578   17616   18406   18410   18414
                    18418   18692   18703   18718   18726   18915   19022   19065   19098   19150   19189   19250
                    19322   19410   19414   19418   19461   19465   19474   19493   19529   19566   19583   19640
                    19719   19723   19733   20099   20124   20135   20156   20192   20195   20203   20445   20458
                    20504   20509   20514   20531   20536   20545   20570   20581   20586   20591   20594   20614
                    20623   20633   20643   20653   20663   20672   20682   20692   20702   20712   20721   20735
                    20739   20746   20761   20816   20820   20901   20914   20930   20950   20962   21102   21112
                    21154   21160   21206   21216   21234   21255   21272   21350   21412   21434   21439   21443
                    21450   21454   21461   21468   21472   21513   21521   21527   21532   21536   21543   21546
                    21585   21589   21600   21608   21615   21621   21651   21655   21662   21669   21678   21682
                    21693   21702   21708   21723   21730   21758   21764   21768   21791   21794   21801   21808
                    21848   21854   21858   21871   21874   21881   21893   21896   21903   21916   21989   22046
                    22447   22458   22627   22643   22674   22722   22773   22789   22991   22994   23050   23053
                    23061   23065   23107   23116   23120   23187   23198   23201   23248   23266   23270   23326
                    23329   23450   23454   23527   23541   23545   23551   23558   23563   23761   23808   23818
                    23826   23835   23839   23877   23885   23889   23928   24015   24103   24196   24251   24318
                    24328   24372   24427   24569   24572   24712   24789   24810   24821   24831   24844   24855
                    24860   24873   25114   25123   25128   25322   25364   25376   25388   25661   25692   25718
                    25807   25869   25873   26189   26198   26306   26333   26338   26341   26593   26645   26671
                    26675   26679   26683   26876   26900   27048   27344   27705   27724   27728   28110   28121
                    28155   28160   28208   28334   28338   28661   28665   28849   28866   29163   29214   29228
                    29238   29276   29339   29356   29366   29370   29398   29579   29615   29645   29680   29684
                    29698   29723   29727   29831   29843   29912   30025   30029   30039   30118   30215   30360
                    30365   30447   30632   30636   30714   30756   30760   30875   30879   30888   30909   30926
                    30931   30968   31126   31134   31245   31286   31291   31301   31305   31337   31349   31527
                    31719   31753   32037   32375   32385   32396   32400   32404   32482   32511   32815   32823
                    32829   32833   32837   33019   33496   33506   33556   33559   33575   33945   33956   34369
                    34432   34447   34451   34489   34510   34557   34616   34673   34704   35027   35107   35111
                    35123   35127   35132   35136   35257   35276   35309   35498   35503   35512   35526   35590
                    35593   35617   35629   35651   35660   35779   35783   35850   35853   35857   35861   35873
```

Cross Reference Listing - Expression Names

	35885	35919	35923	35927	35931	35935	35939	3613.	36248	36293	36304	36316
	36342	36345	36350	36357	36486	36527	36570	3661)	36643	36667	36684	36701
	36707	36712	36715	36719	36729	36736	36745	36759	36763	36782	36785	36796
	36809	36818	36841	36853	36861	36867	36871	36878	36882	36888	36896	36926
	36932	36938	37110	37115	37121	37131	37137	37182	37189	37256	37331	37336
	37346	37357	37365	37375	37380	37388	37400	37483	37492	37563	37663	37679
	37686	37755	37765	37770	37773	37776	37801	37839	37853	37878	37914	37962
	37965	37969	37973	37977	37981	37990	38092	38106	38113	38169	38214	38229
	38279	38493	38563	38566	38579	38655	38659	38722	38772	38776	38807	38918
	39087	39167	39181	39185	39191	39210	39269	39329	39335	39398	39434	39644
	39649	39693	39726	39740	39755	39760	39806	39813	39822	39828	39834	39885
	39925	39932	39949	39960	40039	40050	40061	40066	40071	40096	40185	40208
	40256	40262	40300	40303	40309	40321	40327	40330	40368	40380	40390	40394
	40439	40506	40509	40515	40532	40538	40577	40594	40630	40636	40715	40723
	40775	40782	40858	40996	41032	41070	41108	41181	41265	41362	41385	41490
	41529	41561	41614	41643	41664	41693	41698	41703	41728	41782	41807	41826
	41869	41901	42041	42171	42217	42259	42324	42357	42398	42404	42435	42439
	42443	42465	42493	42508	42559	42581	42598	42612	42626	42655	42670	42683
	42707	42720	42745	42749	42754	42759	42764	42769	42779	42785	42789	42794
	42855	42866	42893	42897	42902	42907	42913	42918	42928	42935	42939	42944
	42959	43001	43007	43034	43038	43043	43048	43054	43059	43076	43080	43084
	43098	43140	43148	43192	43199	43209	43227	43232	43237	44354	44369	44379
	44384	44388	44398	44403	44449	44508	44512	44523	44534	44568	44593	44597
	44604	44608	44619	44630	44694	44698	44709	44714	44751	44782	44786	44805
	44826	44837	44847	44852	44857	44866	44871	44909	44974	44979	45002	45103
	45108	45174	45187	45201	45207	45212	45222	45227	45252	45281	45292	45302
	45306	45310	45320	45325	45361	45375	45385	45390	45394	45404	45409	45448
	45462	45472	45477	45481	45491	45496	45511	45525	45542	45559	45595	45608
	45618	45623	45627	45637	45642	45654	45708	45718	45728	45733	45737	45747
	45752	45767	45775	45786	45795	45823	45834	45844	45849	45854	45863	45868
	45905	45922	45934	45939	45944	45956	45961	45974	46014	46025	46035	46040
	46044	46054	46059	46071	46079	46148	46166	46179	46243	46251	46305	46313
	46391	46407	46416	46432	46441	46457	46475	46492	46569	46576	46583	46590
	46595	46601	46608	46615	46622	46629	46635	46642	46648	46655	46662	46668
	46675	46682	46689	46696	46744	46768	46815	46832	46875	46883	46886	46890
	46893	46895	46908	46911	46919	46922	46926	46947	46952	46990	46993	47000
	47003	47009	47012	47015	47019	47022	47056	47066	47091	47094	47098	47101
	47109	47112	47139	47154	47158	47162	47165	47169	47173	47176	47181	47187
	47197	47201	47208	47211	47215	47219	47222	47227	47237	47241	47248	47251
	47255	47259	47262	47265	47271	47295	47299	47302	47326	47338	47341	47356
	47371	47378	47402	47415	47570	47810						
V001	3588 #	5523	5605	5892	6145	6221	6277	6653	6769	6822	6886	6952
	6987	7021	7162	7389	7398	7416	7426	20487	34570	35451	35507	35543
	35547	35580	35948	36097	36121	36348	36483	36500	36504	36509	36567	36584
	36588	36636	36639	36799	36916	37341	37397	37414	37423	37690	37709	37733
	37804	37807	37813	37819	37827	37841	37857	37918	38035	38149	38160	38267
	38283	38289	38362	38449	39427	39430	39944	39955	40306	40316	40512	40522
	40870	40878	40882	40886	41367	42049	42053	42057	42061	42176	42674	42799
	42949	43088	43204	44374	44393	44489	44505	44585	44601	44675	44691	44763
	44779	44842	44861	44951	44970	45080	45098	45194	45217	45297	45315	45380
	45399	45467	45486	45613	45652	45723	45742	45839	45858	45928	45950	46030
	46049	46159	47184	47190	47230	47268	47274	47278	47284	47419		
V002	3589 #	5564	5569	5579	5596	5630	5672	6765	6783	6791	6806	6840
	6883	6900	6907	6914	7227	8154	8292	8351	8357	8363	8373	8405
	8416	8420	8436	8441	8460	8472	8501	8522	8528	8536	8558	8596

Cross Reference Listing - Expression Names

8606 8620 8653 8685 8695 8709 8732 8770 8780 8814 8824 8858
8869 8916 8932 8942 8986 9045 9050 9072 9149 9154 9159 9164
9319 9329 9336 9341 9502 9515 9519 9523 9527 9670 9687 9721
9727 9731 9737 9743 9750 9756 9765 9775 9782 9791 9835 9840
10679 10694 10699 10704 10733 10754 10767 10770 10777 10804 10817 10822
10825 10831 10875 10996 11107 11166 11177 11235 11240 11245 11262 11353
11383 11387 11473 11482 11489 11658 11663 11672 11678 11691 11699 11706
11756 11777 11790 11899 11910 11917 11930 11935 12035 12042 12745 12985
13064 13169 13248 13320 13573 13578 13582 13596 13630 13646 13653 13657
13678 13693 13712 13716 13774 13800 13956 13974 13992 14242 14317 14343
14635 14697 14702 14723 14740 14751 14769 14774 14804 14809 14816 14822
14854 14859 14949 15073 15079 15088 15095 15104 15110 15115 15120 15188
15192 15207 15220 15322 15489 15496 15737 16069 16235 16242 16502 16742
16755 16771 16782 16812 16827 16868 16877 16894 16933 16950 16957 16997
17246 17531 18227 18241 18245 18279 18291 18364 18372 18672 18679 18711
18723 18734 18868 18873 18878 18883 18888 18893 18898 18903 18927 18931
18935 18939 18943 18947 18951 18955 18971 18976 18981 18986 18991 18996
19001 19005 19010 19109 19113 19117 19121 19125 19129 19293 19299 19391
19451 19607 19611 19615 19619 19623 19627 19631 19635 19645 19659 19664
19669 19674 19679 19684 19689 19693 19698 19728 20197 20208 20211 20230
20240 20244 20249 20253 20258 20262 20266 20270 20275 20279 20284 20288
20293 20297 20301 20305 20310 20314 20319 20323 20328 20332 20336 20340
20345 20425 20434 20441 20455 20469 21037 21041 21147 21166 21362 21458
21464 21483 21539 21550 21612 21618 21625 21705 21712 21726 21734 21805
21814 21900 21907 21920 21999 22004 22017 22058 22093 22498 22557 23046
23103 23183 23258 23307 23311 23475 23521 25031 25095 25714 26212 26896
28172 28298 28594 28619 28623 28633 28810 28824 28828 28832 28836 29694
29703 29707 30002 30046 30127 30132 30253 30267 30273 30277 30281 30286
30290 30346 30356 30374 30380 30388 30468 30510 30532 30536 30611 30620
30659 30710 30747 30853 30857 30861 30865 30924 30959 30963 30974 30978
31249 31253 31257 31261 31745 32041 32442 32463 32498 32844 32848 32926
32930 32965 32970 33011 33031 33041 33953 34437 34495 34610 34663 35751
35756 35760 36174 36361 36777 36813 36858 36875 36885 36900 36911 36929
36935 36942 37353 37361 37369 37994 37998 38002 38006 38011 38040 38374
38457 38461 38483 38487 38572 38616 38622 38664 39205 39245 39264 39295
39731 39745 39817 39890 39896 39899 39903 39935 39939 39980 39984 39987
39991 40180 40218 40232 40241 40267 40319 40333 40445 40906 40910 40914
40934 40938 40942 40991 41001 41007 41021 41027 41038 41045 41059 41065
41076 41083 41097 41103 41114 41121 41135 41193 41206 41215 41221 41225
41247 41260 41271 41282 41500 41539 41568 41573 41587 41595 41623 41628
41657 41679 41683 41742 41763 41772 41821 41831 41837 41854 41860 41886
41892 42192 42196 42200 42211 42222 42243 42309 42336 42346 42376 42384
42393 42428 42479 42503 42538 42546 42551 42566 42571 42621 42631 42645
42736 42815 42860 42883 44349 44364 44432 44473 44477 44485 44493 44497
44501 44551 44572 44580 44589 44658 44662 44670 44679 44683 44687 44747
44759 44767 44771 44775 44799 44821 44934 44965 44992 44998 45012 45022
45063 45075 45094 45121 45126 45356 45370 45425 45457 45517 45570 45604
45659 45678 45683 45916 45978 45987 46171 46176 46187 46349 46871 46879
46903 46915 46944 47053 47059 47062 47144 47147 47332 47361 47382 47386
47389 47960 47975 47979 47988 48008 48014 48018 48022

V004 3590 # 5998 35522
V005 3591 # 39365
V006 3592 # 38146
V007 3593 # 39722 39736
V008 3594 # 39365

Cross Reference Listing - Expression Names

```
V009                3595 #  38146
V01.30              3563 #  36348   36636   36639   38283   40870
V01.31.40           3564 #
V01.31.40.70        3565 #  42799   42949   43088   43204   44374   44393   44489   44505   44585   44601   44675
                    44691   44763   44779   44842   44861   44951   44970   45080   45098   45194   45217   45297
                    45315   45380   45399   45467   45486   45613   45632   45723   45742   45839   45858   45928
                    45950   46030   46049
V01.32.40           3566 #  5523    5605    6277    6653    6769    6987    7021    7162    7389    7398    7416
                    7426    35451   35507   35547   35580   37733   37807   37813   37819   37857   38160   40306
                    40316   40512   40522   46159   47184   47190   47230   47268   47274   47284   47419
V01.32.40.80        3567 #  6145    35543   35948   36097   36121   37690   37709   37804   37827   37841   37918
                    38035   38149   38362   38449   39944   39955   47278
V010                3597 #  35698
V011                3599 #  35585   35599   35612   35625   35703   36679   37810   37816   40250   40365   40441
                    40657   47193   47204   47233   47244
V02.21              3568 #  9149    9154    9159    9164    9670    10770   10875   10996   11107   11235   11240
                    11245   11353   12985   13064   13320   13712   13956   13974   13992   14242   14635   14697
                    14702   14740   14854   14859   15073   15079   15088   15095   15104   15120   15207   16997
                    17246   17531   18227   18364   18372   18672   18679   30659   35751   35756   35760   41595
                    41623   45570   45678   46187   46349   46944   47062   47147   47386   47389   47979
V02.33              3569 #  5564    5569    5579    5596    5630    5672    6765    6783    6791    6806    6840
                    6883    6900    6907    6914    7227    8154    8292    8351    8357    8363    8373    8405
                    8416    8420    8436    8441    8460    8472    8501    8522    8528    8536    8558    8596
                    8606    8620    8653    8685    8695    8709    8732    8770    8780    8814    8824    8858
                    8869    8916    8932    8942    8986    9045    9050    9072    9319    9329    9336    9341
                    9502    9515    9519    9523    9527    9687    9721    9727    9731    9737    9743    9750
                    9756    9765    9775    9782    9791    9835    9840    10679   10694   10699   10704   10733
                    10754   10767   10777   10804   10817   10822   10825   10831   11166   11177   11262   11383
                    11387   11473   11482   11489   11658   11663   11672   11678   11691   11699   11706   11756
                    11777   11790   11899   11910   11917   11930   11935   12035   12042   12745   13169   13248
                    13573   13578   13582   13596   13630   13646   13653   13657   13678   13693   13716   13774
                    13800   14317   14343   14723   14751   14769   14774   14804   14809   14816   14822   14949
                    15110   15115   15188   15192   15220   15322   15489   15496   15737   16069   16235   16242
                    16502   16742   16755   16771   16782   16812   16827   16868   16877   16894   16933   16950
                    16957   18241   18245   18279   18291   18711   18723   18734   19010   19109   19113   19117
                    19121   19125   19129   19293   19299   19391   19451   19645   19698   19728   20197   20208
                    20211   20230   20240   20244   20249   20253   20258   20262   20266   20270   20275   20279
                    20284   20288   20293   20297   20301   20305   20310   20314   20319   20323   20328   20332
                    20336   20340   20345   20425   20434   20441   20455   20469   21037   21041   21147   21166
                    21362   21458   21464   21483   21539   21550   21612   21618   21625   21705   21712   21726
                    21734   21805   21814   21900   21907   21920   21999   22004   22017   22058   22093   22498
                    22557   23046   23103   23183   23258   23307   23311   23475   23521   25031   25095   25714
                    26212   26896   28172   28298   28594   28619   28623   28633   28810   28824   28828   28832
                    28836   29694   29703   29707   30002   30046   30127   30132   30253   30267   30273   30277
                    30281   30286   30290   30346   30356   30374   30380   30388   30468   30510   30532   30536
                    30611   30620   30710   30747   30853   30857   30861   30865   30924   30959   30963   30974
                    30978   31249   31253   31257   31261   31745   32442   32463   32498   32844   32848   32926
                    32930   32965   32970   33011   33031   33041   33953   34437   34495   34610   34663   36174
                    36361   36777   36813   36858   36875   36885   36900   36911   36929   36935   36942   37353
                    37361   37369   37994   37998   38002   38006   38011   38040   38374   38457   38461   38483
                    38487   38572   38664   39731   39745   39817   39890   39896   39899   39903   39935   39939
                    39980   39984   39987   39991   40180   40218   40232   40241   40267   40319   40333   40445
                    40906   40910   40914   40934   40938   40942   40991   41001   41007   41021   41027   41038
                    41045   41059   41065   41076   41083   41097   41103   41114   41121   41135   41193   41206
                    41215   41221   41225   41247   41260   41271   41282   41500   41539   41568   41573   41587
```

Cross Reference Listing - Expression Names

```
                 41628   41657   41679   41683   41742   41763   41772   41821   41831   41837   41854   41860
                 41886   41892   42192   42196   42200   42211   42222   42243   42309   42376   42384   42393
                 42538   42546   42551   42566   42571   42621   42631   42736   42815   42860   42883   44349
                 44432   44473   44477   44485   44493   44497   44501   44551   44572   44580   44589   44658
                 44662   44670   44679   44683   44687   44747   44759   44767   44771   44775   44799   44821
                 44934   44965   44992   44998   45012   45022   45063   45075   45094   45121   45126   45356
                 45425   45517   45659   45683   45978   45987   46171   46176   46871   46879   47053   47059
                 47144   47332   47361   47382   47960   47975   47988   48008   48014   48018   48022
V02.33.41        3570 #  18868   18873   18878   18883   18888   18893   18898   18903   18927   18931   18935
                 18939   18943   18947   18951   18955   18971   18976   18981   18986   18991   18996   19001
                 19005   19607   19611   19615   19619   19623   19627   19631   19635   19659   19664   19669
                 19674   19679   19684   19689   19693   32041   38616   38622   39205   39245   39264   39295
                 42336   42346   42428   42479   42503   42645   44364   45370   45457   45604   45916   46903
                 46915
V020             3602 #  5564    5569    5579    5596    5630    5672    6900    6907    6914    7227    8292
                 8351    8357    8363    8373    8405    8416    8420    8436    8441    8460    8472    8501
                 8522    8528    8536    8558    8596    8606    8620    8653    8685    8695    8709    8732
                 8770    8780    8814    8824    8858    8869    8916    8932    8942    8986    9045    9050
                 9072    9319    9329    9336    9341    9502    9515    9519    9523    9527    9687    9721
                 9727    9731    9737    9743    9750    9756    9765    9775    9782    9835    9840    10694
                 10699   10704   10754   10767   10777   10822   10831   11262   11482   11489   11672   11678
                 11691   11699   11706   11777   11790   11899   11910   11917   11930   11935   13169   13248
                 13774   13800   14317   14343   14723   14751   14769   14774   14804   14809   14816   14822
                 15322   15489   15496   16742   16755   16771   16782   16812   16827   16868   16877   16894
                 16933   16950   16957   18241   18245   18279   18291   18711   18723   18734   19109   19113
                 19117   19121   19125   19129   19293   19299   19391   19451   19645   19728   20197   20230
                 20240   20249   20258   20266   20275   20284   20293   20301   20310   20319   20328   20336
                 20345   20425   20434   20441   20455   20469   21037   21041   21147   21362   21458   21464
                 21483   21539   21550   21612   21618   21625   21705   21712   21726   21734   21805   21814
                 21900   21907   21920   21999   22004   22017   22058   22093   23046   23103   23183   23258
                 23307   23311   23475   23521   25714   26212   26896   28172   28298   28594   28619   28623
                 28633   28810   28824   28828   28832   28836   29694   29703   29707   30002   30046   30127
                 30132   30267   30273   30286   30290   30356   30374   30380   30388   30532   30536   30611
                 30620   30710   30747   30853   30857   30861   30865   30924   30959   30963   30974   30978
                 31745   32844   32848   32926   32930   32965   32970   33011   33031   33041   33953   34437
                 34495   36361   36777   36813   36858   36875   36885   36900   36911   36929   36935   36942
                 37353   37361   37369   37994   37998   38002   38006   38011   38040   38483   38487   38572
                 38664   39674   39678   39684   39688   39731   39745   39817   40180   40218   40232   40991
                 41001   41007   41027   41038   41045   41065   41076   41083   41103   41114   41121   41193
                 41215   41260   41271   41282   41500   41539   41568   41573   41628   41657   41763   41772
                 41821   41831   41837   41854   41860   41886   41892   42211   42222   42243   42309   42376
                 42384   42393   42538   42546   42551   42566   42571   42621   42631   46171   46176   46871
                 46879   47053   47059   47144   47332   47361   47382
V021             3603 #  5324    5328    5335    5940    5948    9149    9154    9159    9164    9170    9175
                 9180    9185    9191    9195    9199    9203    9209    9214    9219    9224    9230    9234
                 9238    9242    9554    9565    9581    9591    9596    9600    9608    9614    9619    9625
                 9630    9634    9642    9648    9653    9670    10770   10875   10996   11107   11235   11240
                 11245   11353   12852   12985   13064   13081   13320   13712   13956   13974   13992   14209
                 14220   14235   14242   14255   14635   14673   14697   14702   14740   14854   14859   15073
                 15079   15088   15095   15104   15120   15207   15783   16997   17246   17531   18227   18364
                 18372   18672   18679   30659   35562   35567   35751   35756   35760   38146   41595   41623
                 45570   45678   46187   46349   46944   47062   47147   47386   47389   47979
V022             3604 #  5547    5622    5687    5733    5754    5791    5800    5818    6779    6787    6796
                 8645    15719   15760   16021   16035   16219   16611   16618   16643   17010   17014   17020
                 17041   17081   17091   17261   17266   17286   17557   18410   18414   18418   18427   18439
```

Cross Reference Listing - Expression Names

	18451	18695	18715	19042	19053	19077	19540	19546	19739	20112	20118	20536
	20581	20586	20591	20614	20623	20633	20643	20653	20663	20672	20682	20692
	20702	20712	20721	20761	21272	21446	21527	21532	21659	21780	21861	21878
	21965	22050	23757	23804	24202	24206	24481	26345	29398	29843	29912	30118
	30215	30451	31074	32408	32412	32435	32456	32475	32491	33500	33515	34380
	36139	36324	36688	36697	36712	36719	36736	36739	36759	36763	36767	36788
	36818	36871	36882	37213	37219	38286	39335	39338	39649	39653	40313	40518
	40581	40600	40633	42604	42652	42716	42821	42831	42847	42967	42977	42993
	43169	43216	43243	43252	44416	44426	44438	44444	44456	44539	44545	44635
	44641	44724	44730	44884	44891	44898	44904	44915	45027	45040	45131	45138
	45240	45247	45338	45420	45511	45525	45542	45548	45554	45565	45654	45664
	45767	45775	45781	45790	45802	45881	45974	46071	46079	46085	46768	47150
	47197	47237	47346	47366								
V023	3605 #	5888	5913	5920	6007	35132	35698	36966	37798	38888	38902	40076
	40081	40096	40159	40164	40185	40208	40422	40427	40643			
V024	3607 #	20156	20553	21108	21210	21218	21594	21675	21690	21798	21885	29209
	29620	35632	36125	36470	36488	36554	36572	38419	39796	40028	40042	40201
	40371	40383	40557	40568	40577	40586	40594	40622	40630	40657	40687	40751
	40810	40854										
V025	3608 #	20156	20553	21108	21210	21218	21594	21675	21690	21798	21885	29209
	29620	35632	36125	36470	36488	36554	36572	38419	39796	40028	40042	40201
	40371	40383	40557	40568	40577	40586	40594	40622	40630	40657	40687	40751
	40810	40854										
V030	3611 #	5358	5422	5455	5459	5463	5519	5526	5547	5553	5611	5636
	5659	5667	5676	5682	5699	5712	5726	5733	5750	5762	5791	5796
	5800	5882	5888	5907	5920	5927	5934	6122	6139	6142	6236	6437
	6657	6686	6702	6746	6872	6876	6969	6974	7014	7027	7042	7045
	7184	7190	7220	8128	8132	8136	8139	8411	8455	8467	8532	9055
	9063	9498	9511	9760	9770	9778	9788	9805	11923	12580	12642	14763
	14791	14795	14799	15340	15551	15714	15719	15760	15992	16021	16035	16141
	16219	16284	16294	16534	16611	16618	16643	16776	16888	16927	16943	17010
	17014	17020	17041	17076	17081	17091	17160	17169	17173	17177	17261	17266
	17286	17470	17488	17552	17578	17616	18406	18410	18414	18418	18692	18703
	18718	18726	18915	19022	19065	19098	19150	19189	19250	19322	19410	19414
	19418	19461	19465	19474	19493	19529	19566	19583	19640	19719	19723	19733
	20099	20124	20135	20192	20195	20203	20445	20458	20504	20509	20514	20531
	20536	20545	20570	20581	20586	20591	20594	20614	20623	20633	20643	20653
	20663	20672	20682	20692	20702	20712	20721	20735	20739	20746	20761	20816
	20820	20901	20914	20930	20962	21102	21112	21154	21160	21206	21216	21234
	21255	21272	21434	21439	21443	21450	21454	21461	21468	21472	21513	21521
	21527	21532	21536	21543	21546	21585	21589	21600	21608	21615	21621	21651
	21655	21662	21669	21678	21682	21693	21702	21708	21723	21730	21758	21764
	21768	21791	21794	21801	21808	21848	21854	21858	21871	21874	21881	21893
	21896	21903	21916	21989	22046	22447	22458	22627	22643	22674	22722	22773
	22789	22991	22994	23050	23053	23061	23065	23107	23116	23120	23187	23198
	23201	23248	23266	23270	23326	23329	23450	23454	23527	23541	23545	23551
	23558	23563	23761	23808	23818	23826	23835	23839	23877	23885	23889	23928
	24015	24103	24196	24251	24318	24328	24372	24427	24569	24572	24712	24789
	24810	24821	24831	24844	24855	24860	24873	25114	25123	25128	25322	25364
	25376	25388	25661	25692	25718	25807	25869	25873	26189	26198	26306	26333
	26338	26341	26593	26645	26671	26675	26679	26683	26876	26900	27048	27344
	27705	27724	27728	28110	28121	28155	28160	28208	28334	28338	28661	28665
	28849	28866	29163	29214	29228	29238	29276	29339	29356	29366	29370	29398
	29579	29615	29645	29680	29684	29698	29723	29727	29831	29843	29912	30025
	30039	30118	30215	30360	30365	30447	30632	30636	30714	30756	30760	30875

Cross Reference Listing - Expression Names

	30879	30888	30909	30926	30931	30968	31126	31134	31245	31286	31291	31301
	31305	31337	31349	31527	31719	31753	32037	32375	32385	32396	32400	32404
	32482	32511	32815	32823	32829	32833	32837	33019	33496	33506	33556	33575
	33945	33956	34369	34432	34447	34451	34489	34510	34557	34616	34673	35027
	35107	35111	35123	35127	35132	35136	35257	35276	35309	35498	35503	35512
	35526	35590	35593	35617	35629	35783	35850	35853	35857	35861	35873	35885
	35919	35923	35927	35931	35935	35939	36136	36293	36304	36316	36342	36348
	36357	36486	36527	36570	36636	36639	36643	36667	36684	36701	36707	36712
	36715	36719	36729	36736	36745	36759	36763	36782	36785	36809	36818	36841
	36853	36861	36867	36871	36878	36882	36888	36926	36932	36938	37110	37115
	37121	37131	37137	37256	37331	37346	37357	37365	37375	37380	37388	37400
	37483	37492	37563	37663	37679	37686	37755	37765	37770	37773	37776	37801
	37839	37853	37878	37914	37962	37965	37969	37973	37977	37981	37990	38092
	38106	38113	38169	38214	38229	38279	38283	38493	38563	38566	38579	38655
	38659	38722	38772	38776	38807	38918	39087	39167	39181	39185	39191	39210
	39269	39329	39335	39365	39398	39434	39644	39649	39693	39726	39740	39755
	39760	39806	39813	39822	39828	39834	39885	39925	39932	39949	39960	40039
	40050	40061	40066	40071	40096	40185	40208	40256	40262	40300	40303	40309
	40321	40327	40330	40368	40380	40390	40394	40439	40506	40509	40515	40532
	40538	40577	40594	40630	40636	40715	40723	40775	40782	40870	40996	41032
	41070	41108	41181	41265	41490	41529	41614	41728	41782	41807	41826	41869
	41901	42217	42259	42324	42357	42398	42404	42435	42439	42443	42465	42493
	42559	42581	42598	42612	42626	42655	42707	42720	42745	42749	42754	42759
	42764	42769	42779	42785	42789	42794	42855	42866	42893	42897	42902	42907
	42913	42918	42928	42935	42939	42944	42959	43001	43007	43034	43038	43043
	43048	43054	43059	43076	43080	43084	43098	43140	43148	43192	43199	43209
	43227	43232	43237	44354	44369	44379	44384	44388	44398	44403	44449	44508
	44512	44523	44534	44568	44593	44597	44604	44608	44619	44630	44694	44698
	44709	44714	44751	44782	44786	44805	44826	44837	44847	44852	44857	44866
	44871	44909	44974	44979	45002	45103	45108	45174	45187	45201	45207	45212
	45222	45227	45252	45281	45292	45302	45306	45310	45320	45325	45361	45375
	45385	45390	45394	45404	45409	45448	45462	45472	45477	45481	45491	45496
	45511	45525	45542	45559	45595	45608	45618	45623	45627	45637	45642	45654
	45708	45718	45728	45733	45737	45747	45752	45767	45775	45786	45795	45823
	45834	45844	45849	45854	45863	45868	45905	45922	45934	45939	45944	45956
	45961	45974	46014	46025	46035	46040	46044	46054	46059	46071	46079	46148
	46166	46179	46243	46251	46305	46313	46391	46407	46416	46432	46441	46457
	46475	46492	46569	46576	46583	46590	46595	46601	46608	46615	46622	46629
	46635	46642	46648	46655	46662	46668	46675	46682	46689	46696	46744	46768
	46815	46832	46875	46883	46886	46890	46893	46895	46908	46911	46919	46922
	46926	46947	46952	46990	46993	47000	47003	47009	47012	47015	47019	47022
	47056	47066	47091	47094	47098	47101	47109	47112	47139	47154	47158	47162
	47165	47169	47173	47176	47181	47187	47197	47201	47208	47211	47215	47219
	47222	47227	47237	47241	47248	47251	47255	47259	47262	47265	47271	47295
	47299	47302	47326	47338	47341	47356	47371	47378	47402	47415	47810	
V031	3612 #	42799	42949	43088	43204	44374	44393	44489	44505	44585	44601	44675
	44691	44763	44779	44842	44861	44951	44970	45080	45098	45194	45217	45297
	45315	45380	45399	45467	45486	45613	45632	45723	45742	45839	45858	45928
	45950	46030	46049									
V032	3613 #	5523	5605	6145	6277	6653	6769	6987	7021	7162	7389	7398
	7416	7426	35451	35507	35543	35547	35580	35948	36097	36121	37690	37709
	37733	37804	37807	37813	37819	37827	37841	37857	37918	38035	38149	38160
	38362	38449	39944	39955	40306	40316	40512	40522	46159	47184	47190	47230
	47268	47274	47278	47284	47419							
V033	3614 #	5564	5569	5579	5596	5630	5672	5998	6765	6783	6791	6806

Cross Reference Listing - Expression Names

6840	6883	6900	6907	6914	7227	8154	8292	8351	8357	8363	8373
8405	8416	8420	8436	8441	8460	8472	8501	8522	8528	8536	8558
8596	8606	8620	8653	8685	8695	8709	8732	8770	8780	8814	8824
8858	8869	8916	8932	8942	8986	9045	9050	9072	9319	9329	9336
9341	9502	9515	9519	9523	9527	9687	9721	9727	9731	9737	9743
9750	9756	9765	9775	9782	9791	9835	9840	10679	10694	10699	10704
10733	10754	10767	10777	10804	10817	10822	10825	10831	11166	11177	11262
11383	11387	11473	11482	11489	11658	11663	11672	11678	11691	11699	11706
11756	11777	11790	11899	11910	11917	11930	11935	12035	12042	12745	13169
13248	13573	13578	13582	13596	13630	13646	13653	13657	13678	13693	13716
13774	13800	14317	14343	14723	14751	14769	14774	14804	14809	14816	14822
14949	15110	15115	15188	15192	15220	15322	15489	15496	15737	16069	16235
16242	16502	16742	16755	16771	16782	16812	16827	16868	16877	16894	16933
16950	16957	18241	18245	18279	18291	18311	18723	18734	18868	18873	18878
18883	18888	18893	18898	18903	18927	18931	18935	18939	18943	18947	18951
18955	18971	18976	18981	18986	18991	18996	19001	19005	19010	19109	19113
19117	19121	19125	19129	19293	19299	19391	19451	19607	19611	19615	19619
19623	19627	19631	19635	19645	19659	19664	19669	19674	19679	19684	19689
19693	19698	19728	20197	20208	20211	20230	20240	20244	20249	20253	20258
20262	20266	20270	20275	20279	20284	20288	20293	20297	20301	20305	20310
20314	20319	20323	20328	20332	20336	20340	20345	20425	20434	20441	20455
20469	21037	21041	21147	21166	21362	21458	21464	21483	21539	21550	21612
21618	21625	21705	21712	21726	21734	21805	21814	21900	21907	21920	21999
22004	22017	22058	22093	22498	22557	23046	23103	23183	23258	23307	23311
23475	23521	25031	25095	25714	26212	26896	28172	28298	28594	28619	28623
28633	28810	28824	28828	28832	28836	29694	29703	29707	30002	30046	30127
30132	30253	30267	30273	30277	30281	30286	30290	30346	30356	30374	30380
30388	30468	30510	30532	30536	30611	30620	30710	30747	30853	30857	30861
30865	30924	30959	30963	30974	30978	31249	31253	31257	31261	31745	32041
32442	32463	32498	32844	32848	32926	32930	32965	32970	33011	33031	33041
33953	34437	34495	34610	34663	35522	36174	36361	36777	36813	36858	36875
36885	36900	36911	36929	36935	36942	37353	37361	37369	37994	37998	38002
38006	38011	38040	38374	38457	38461	38483	38487	38572	38616	38622	38664
39205	39245	39264	39295	39731	39745	39817	39890	39896	39899	39903	39935
39939	39980	39984	39987	39991	40180	40218	40232	40241	40267	40319	40333
40445	40906	40910	40914	40934	40938	40942	40991	41001	41007	41021	41027
41038	41045	41059	41065	41076	41083	41097	41103	41114	41121	41135	41193
41206	41215	41221	41225	41247	41260	41271	41282	41500	41539	41568	41573
41587	41628	41657	41679	41683	41742	41763	41772	41821	41831	41837	41854
41860	41886	41892	42192	42196	42200	42211	42222	42243	42309	42336	42346
42376	42384	42393	42428	42479	42503	42538	42546	42551	42566	42571	42621
42631	42645	42736	42815	42860	42883	44349	44364	44432	44473	44477	44485
44493	44497	44501	44551	44572	44580	44589	44658	44662	44670	44679	44683
44687	44747	44759	44767	44771	44775	44799	44821	44934	44965	44992	44998
45012	45022	45063	45075	45094	45121	45126	45356	45370	45425	45457	45517
45604	45659	45683	45916	45978	45987	46171	46176	46871	46879	46903	46915
47053	47059	47144	47332	47361	47382	47960	47975	47988	48008	48014	48018
48022											

V034	3615 #	35585	35599	35612	35625	35703	36679	37810	37816	40250	40365	40441
	40657	47193	47204	47233	47244							
V035	3616 #	35698										
V036	3617 #	38146										
V037	3618 #	39722	39736									
V038	3619 #	39722	39736									
V04.33	3571 #	5998	35522									

Cross Reference Listing - Expression Names

V04.34 3572 #
V04.35 3573 #
V040 3622 # 5523 5605 6145 6277 6653 6769 6987 7021 7162 7389 7398
7416 7426 35451 35507 35543 35547 35580 35948 36097 36121 37690 37709
37733 37804 37807 37813 37819 37827 37841 37857 37918 38035 38146 38149
38160 38362 38449 39944 39955 40306 40316 40512 40522 42799 42949 43088
43204 44374 44393 44489 44505 44585 44601 44675 44691 44763 44779 44842
44861 44951 44970 45080 45098 45194 45217 45297 45315 45380 45399 45467
45486 45613 45632 45723 45742 45839 45858 45928 45950 46030 46049 46159
47184 47190 47230 47268 47274 47278 47284 47419
V041 3623 # 5358 5422 5455 5459 5463 5667 5676 5882 6122 6139 6437
6657 6702 6969 7027 7042 7045 7190 15340 16294 17470 17488 17578
17616 18868 18873 18878 18883 18888 18893 18898 18903 18915 18927 18931
18935 18939 18943 18947 18951 18955 18971 18976 18981 18986 18991 18996
19001 19005 19022 19098 19189 19250 19322 19410 19414 19418 19461 19465
19474 19493 19566 19583 19607 19611 19615 19619 19623 19627 19631 19635
19659 19664 19669 19674 19679 19684 19689 19693 19719 20458 20594 20914
20962 22458 22627 22643 22674 22722 22789 22994 23050 23053 23107 23187
23450 23454 23527 23551 23761 23808 23818 23877 23928 24015 24103 24251
24318 24328 24372 24427 24569 24712 24789 24844 24860 24873 25123 25376
25388 26189 26198 26306 26333 27724 27728 28849 28866 29228 29238 29339
29370 30025 31126 31245 31286 31291 31301 31305 32037 32041 33556 33575
33956 34557 34616 34673 35593 35617 36527 36715 36841 37256 37346 37380
37388 37755 37776 38106 38616 38622 38807 39205 39210 39245 39264 39269
39295 39434 39644 39755 40256 40390 40996 41032 41070 41108 41181 41265
41490 41529 41614 41728 41782 41807 41826 41869 41901 42217 42259 42324
42336 42346 42357 42398 42428 42435 42439 42443 42465 42479 42493 42503
42626 42645 42655 42855 43001 43080 43192 44364 44751 45002 45370 45457
45604 45916 46243 46251 46305 46313 46391 46407 46416 46432 46441 46457
46475 46492 46569 46576 46583 46590 46595 46601 46608 46615 46622 46629
46635 46642 46648 46655 46662 46668 46675 46682 46689 46696 46744 46886
46893 46895 46903 46911 46915 46922 46926 46947 46993 47003 47012 47022
47066 47094 47101 47112 47154 47165 47176 47197 47211 47222 47237 47251
47262 47295 47302 47356 47378
V05.30 3574 # 39365
V050 3626 # 9170 9175 9180 9185 9191 9195 9199 9203 9209 9214 9219
9224 9230 9234 9238 9242 9454 9465 9471 9554 9565 9581 9591
9596 9600 9608 9614 9619 9625 9630 9634 9642 9648 9653 12852
13081 14209 14220 14235 14255 14462 14480 14673 15783 42119 42125 42134
V051 3627 # 9170 9175 9180 9185 9191 9195 9199 9203 9209 9214 9219
9224 9230 9234 9238 9242 9554 9565 9581 9591 9596 9600 9608
9614 9619 9625 9630 9634 9642 9648 9653 12852 13081 14209 14220
14235 14255 14673 15783
V052 3630 # 9170 9175 9180 9185 9191 9195 9199 9203 9209 9214 9219
9224 9230 9234 9238 9242 9554 9565 9581 9591 9596 9600 9608
9614 9619 9625 9630 9634 9642 9648 9653 12852 13081 14209 14220
14235 14255 14673 15783
V053 3631 # 9170 9175 9180 9185 9191 9195 9199 9203 9209 9214 9219
9224 9230 9234 9238 9242 9554 9565 9581 9591 9596 9600 9608
9614 9619 9625 9630 9634 9642 9648 9653 12852 13081 14209 14220
14235 14255 14673 15783
V054 3632 # 9170 9175 9180 9185 9191 9195 9199 9203 9209 9214 9219
9224 9230 9234 9238 9242 9554 9565 9581 9591 9596 9600 9608
9614 9619 9625 9630 9634 9642 9648 9653 12852 13081 14209 14220
14235 14255 14673 15783

Cross Reference Listing - Expression Names

V06.36.40
3575 # 38146
V060
3634 # 9850 9857 9864 9871 9878 9885 9892 9898 9905 9912 9919
9926 9933 9940 9947 9953 9960 9967 9974 9981 9988 9995 10002
10008 10015 10022 10029 10036 10043 10050 10057 10063 10070 10077 10084
10091 10098 10105 10112 10118 10125 10132 10139 10146 10153 10160 10167
10173 10180 10187 10194 10201 10208 10215 10222 10228 10235 10242 10249
10256 10263 10270 10277 10283 10290 10297 10304 10311 10318 10325 10332
10338 10345 10352 10359 10366 10373 10380 10387 10393 10400 10407 10414
10421 10428 10435 10442 10448 10455 10462 10469 10476 10483 10490 10497
10503 10510 17646 17653 17660 17667 17674 17681 17692 17698 17705 17712
17719 17726 17733 17740 17747 17753 17760 17767 17774 17781 17788 17795
17802 17808 17815 17822 17829 17836 17843 17850 17857 17863 17870 17881
17888 17895 17902 17909 17915 17922 17929 17940 17947 17954 17961 17967
17974 18778 18785 18792 18799 18806 18813 18820 19745 19752 19759 19766
19773 19780 19787 19793 19800 19807 19814 19821 19828 19835 19842 19848
19857 19866 19873 19880 19887 19894 19900 19907 19916 19925 19932 19939
19946 19952 19959 19966 19973 19980 19987 19994 20001 20007 20014 20826
20835 20844 22109 22116 22123 22130 22137 22144 22151 22157 22164 22171
22178 22185 22192 22199 22206 22212 25413 25420 25427 25434 25441 25448
25455 25461 25468 25475 25482 33252 33259 33266 33273 33280 33287 33294
33300 33307 33314 33321 33328 33335 33342 33348 33355 34773 36367 36374
36381 36388 36395 36402 36984 36994 37004 37014 39514 39521 39528 39535
39542 39549 39556 39562 39569
V061
3635 # 9850 9857 9864 9871 9878 9885 9892 9898 9905 9912 9919
9926 9933 9940 9947 9953 9960 9967 9974 9981 9988 9995 10002
10008 10015 10022 10029 10036 10043 10050 10057 10063 10070 10077 10084
10091 10098 10105 10112 10118 10125 10132 10139 10146 10153 10160 10167
10173 10180 10187 10194 10201 10208 10215 10222 10228 10235 10242 10249
10256 10263 10270 10277 10283 10290 10297 10304 10311 10318 10325 10332
10338 10345 10352 10359 10366 10373 10380 10387 10393 10400 10407 10414
10421 10428 10435 10442 10448 10455 10462 10469 10476 10483 10490 10497
10503 10510 17646 17653 17660 17667 17674 17681 17692 17698 17705 17712
17719 17726 17733 17740 17747 17753 17760 17767 17774 17781 17788 17795
17802 17808 17815 17822 17829 17836 17843 17850 17857 17863 17870 17881
17888 17895 17902 17909 17915 17922 17929 17940 17947 17954 17961 17967
17974 18778 18785 18792 18799 18806 18813 18820 19745 19752 19759 19766
19773 19780 19787 19793 19800 19807 19814 19821 19828 19835 19842 19848
19857 19866 19873 19880 19887 19894 19900 19907 19916 19925 19932 19939
19946 19952 19959 19966 19973 19980 19987 19994 20001 20007 20014 20826
20835 20844 22109 22116 22123 22130 22137 22144 22151 22157 22164 22171
22178 22185 22192 22199 22206 22212 25413 25420 25427 25434 25441 25448
25455 25461 25468 25475 25482 33252 33259 33266 33273 33280 33287 33294
33300 33307 33314 33321 33328 33335 33342 33348 33355 34773 36367 36374
36381 36388 36395 36402 36984 36994 37004 37014 39514 39521 39528 39535
39542 39549 39556 39562 39569
V062
3636 # 9853 9860 9867 9874 9881 9888 9895 9901 9908 9915 9922
9929 9936 9943 9950 9956 9963 9970 9977 9984 9991 9998 10005
10011 10018 10025 10032 10039 10046 10053 10060 10066 10073 10080 10087
10094 10101 10108 10115 10121 10128 10135 10142 10149 10156 10163 10170
10176 10183 10190 10197 10204 10211 10218 10225 10231 10238 10245 10252
10259 10266 10273 10280 10286 10293 10300 10307 10314 10321 10328 10335
10341 10348 10355 10362 10369 10376 10383 10390 10396 10403 10410 10417
10424 10431 10438 10445 10451 10458 10465 10472 10479 10486 10493 10500
10506 10513 17649 17656 17663 17670 17677 17684 17695 17701 17708 17715
17722 17729 17736 17743 17750 17756 17763 17770 17777 17784 17791 17798

Cross Reference Listing - Expression Names

17805 17811 17818 17825 17832 17839 17846 17853 17860 17866 17873 17884
17891 17898 17905 17912 17918 17925 17932 17943 17950 17957 17964 17970
17977 18781 18788 18795 18802 18809 18816 18823 19748 19755 19762 19769
19776 19783 19790 19796 19803 19810 19817 19824 19831 19838 19845 19851
19860 19869 19876 19883 19890 19897 19903 19910 19919 19928 19935 19942
19949 19955 19962 19969 19976 19983 19990 19997 20004 20010 20017 20829
20838 20847 22112 22119 22126 22133 22140 22147 22154 22160 22167 22174
22181 22188 22195 22202 22209 22215 25416 25423 25430 25437 25444 25451
25458 25464 25471 25478 25485 33255 33262 33269 33276 33283 33290 33297
33303 33310 33317 33324 33331 33338 33345 33351 33358 34776 36370 36377
36384 36391 36398 36405 36987 36997 37007 37017 39517 39524 39531 39538
39545 39552 39559 39565 39572

V063 3637 # 9853 9860 9867 9874 9881 9888 9895 9901 9908 9915 9922
9929 9936 9943 9950 9956 9963 9970 9977 9984 9991 9998 10005
10011 10018 10025 10032 10039 10046 10053 10060 10066 10073 10080 10087
10094 10101 10108 10115 10121 10128 10135 10142 10149 10156 10163 10170
10176 10183 10190 10197 10204 10211 10218 10225 10231 10238 10245 10252
10259 10266 10273 10280 10286 10293 10300 10307 10314 10321 10328 10335
10341 10348 10355 10362 10369 10376 10383 10390 10396 10403 10410 10417
10424 10431 10438 10445 10451 10458 10465 10472 10479 10486 10493 10500
10506 10513 17649 17656 17663 17670 17677 17684 17695 17701 17708 17715
17722 17729 17736 17743 17750 17756 17763 17770 17777 17784 17791 17798
17805 17811 17818 17825 17832 17839 17846 17853 17860 17866 17873 17884
17891 17898 17905 17912 17918 17925 17932 17943 17950 17957 17964 17970
17977 18781 18788 18795 18802 18809 18816 18823 19748 19755 19762 19769
19776 19783 19790 19796 19803 19810 19817 19824 19831 19838 19845 19851
19860 19869 19876 19883 19890 19897 19903 19910 19919 19928 19935 19942
19949 19955 19962 19969 19976 19983 19990 19997 20004 20010 20017 20829
20838 20847 22112 22119 22126 22133 22140 22147 22154 22160 22167 22174
22181 22188 22195 22202 22209 22215 25416 25423 25430 25437 25444 25451
25458 25464 25471 25478 25485 33255 33262 33269 33276 33283 33290 33297
33303 33310 33317 33324 33331 33338 33345 33351 33358 34776 36370 36377
36384 36391 36398 36405 36987 36997 37007 37017 39517 39524 39531 39538
39545 39552 39559 39565 39572

V064 3638 # 9853 9860 9867 9874 9881 9888 9895 9901 9908 9915 9922
9929 9936 9943 9950 9956 9963 9970 9977 9984 9991 9998 10005
10011 10018 10025 10032 10039 10046 10053 10060 10066 10073 10080 10087
10094 10101 10108 10115 10121 10128 10135 10142 10149 10156 10163 10170
10176 10183 10190 10197 10204 10211 10218 10225 10231 10238 10245 10252
10259 10266 10273 10280 10286 10293 10300 10307 10314 10321 10328 10335
10341 10348 10355 10362 10369 10376 10383 10390 10396 10403 10410 10417
10424 10431 10438 10445 10451 10458 10465 10472 10479 10486 10493 10500
10506 10513 17649 17656 17663 17670 17677 17684 17695 17701 17708 17715
17722 17729 17736 17743 17750 17756 17763 17770 17777 17784 17791 17798
17805 17811 17818 17825 17832 17839 17846 17853 17860 17866 17873 17884
17891 17898 17905 17912 17918 17925 17932 17943 17950 17957 17964 17970
17977 18781 18788 18795 18802 18809 18816 18823 19748 19755 19762 19769
19776 19783 19790 19796 19803 19810 19817 19824 19831 19838 19845 19851
19860 19869 19876 19883 19890 19897 19903 19910 19919 19928 19935 19942
19949 19955 19962 19969 19976 19983 19990 19997 20004 20010 20017 20829
20838 20847 22112 22119 22126 22133 22140 22147 22154 22160 22167 22174
22181 22188 22195 22202 22209 22215 25416 25423 25430 25437 25444 25451
25458 25464 25471 25478 25485 33255 33262 33269 33276 33283 33290 33297
33303 33310 33317 33324 33331 33338 33345 33351 33358 34776 36370 36377
36384 36391 36398 36405 36987 36997 37007 37017 39517 39524 39531 39538

	39545	39552	39559	39565	39572							
V065	3639 #	9853	9860	9867	9874	9881	9888	9895	9901	9908	9915	9922
	9929	9936	9943	9950	9956	9963	9970	9977	9984	9991	9998	10005
	10011	10018	10025	10032	10039	10046	10053	10060	10066	10073	10080	10087
	10094	10101	10108	10115	10121	10128	10135	10142	10149	10156	10163	10170
	10176	10183	10190	10197	10204	10211	10218	10225	10231	10238	10245	10252
	10259	10266	10273	10280	10286	10293	10300	10307	10314	10321	10328	10335
	10341	10348	10355	10362	10369	10376	10383	10390	10396	10403	10410	10417
	10424	10431	10438	10445	10451	10458	10465	10472	10479	10486	10493	10500
	10506	10513	17649	17656	17663	17670	17677	17684	17695	17701	17708	17715
	17722	17729	17736	17743	17750	17756	17763	17770	17777	17784	17791	17798
	17805	17811	17818	17825	17832	17839	17846	17853	17860	17866	17873	17884
	17891	17898	17905	17912	17918	17925	17932	17943	17950	17957	17964	17970
	17977	18781	18788	18795	18802	18809	18816	18823	19748	19755	19762	19769
	19776	19783	19790	19796	19803	19810	19817	19824	19831	19838	19845	19851
	19860	19869	19876	19883	19890	19897	19903	19910	19919	19928	19935	19942
	19949	19955	19962	19969	19976	19983	19990	19997	20004	20010	20017	20829
	20838	20847	22112	22119	22126	22133	22140	22147	22154	22160	22167	22174
	22181	22188	22195	22202	22209	22215	25416	25423	25430	25437	25444	25451
	25458	25464	25471	25478	25485	33255	33262	33269	33276	33283	33290	33297
	33303	33310	33317	33324	33331	33338	33345	33351	33358	34776	36370	36377
	36384	36391	36398	36405	36987	36997	37007	37017	39517	39524	39531	39538
	39545	39552	39559	39565	39572							
V066	3640 #	9853	9860	9867	9874	9881	9888	9895	9901	9908	9915	9922
	9929	9936	9943	9950	9956	9963	9970	9977	9984	9991	9998	10005
	10011	10018	10025	10032	10039	10046	10053	10060	10066	10073	10080	10087
	10094	10101	10108	10115	10121	10128	10135	10142	10149	10156	10163	10170
	10176	10183	10190	10197	10204	10211	10218	10225	10231	10238	10245	10252
	10259	10266	10273	10280	10286	10293	10300	10307	10314	10321	10328	10335
	10341	10348	10355	10362	10369	10376	10383	10390	10396	10403	10410	10417
	10424	10431	10438	10445	10451	10458	10465	10472	10479	10486	10493	10500
	10506	10513	17649	17656	17663	17670	17677	17684	17695	17701	17708	17715
	17722	17729	17736	17743	17750	17756	17763	17770	17777	17784	17791	17798
	17805	17811	17818	17825	17832	17839	17846	17853	17860	17866	17873	17884
	17891	17898	17905	17912	17918	17925	17932	17943	17950	17957	17964	17970
	17977	18781	18788	18795	18802	18809	18816	18823	19748	19755	19762	19769
	19776	19783	19790	19796	19803	19810	19817	19824	19831	19838	19845	19851
	19860	19869	19876	19883	19890	19897	19903	19910	19919	19928	19935	19942
	19949	19955	19962	19969	19976	19983	19990	19997	20004	20010	20017	20829
	20838	20847	22112	22119	22126	22133	22140	22147	22154	22160	22167	22174
	22181	22188	22195	22202	22209	22215	25416	25423	25430	25437	25444	25451
	25458	25464	25471	25478	25485	33255	33262	33269	33276	33283	33290	33297
	33303	33310	33317	33324	33331	33338	33345	33351	33358	34776	36370	36377
	36384	36391	36398	36405	36987	36997	37007	37017	39517	39524	39531	39538
	39545	39552	39559	39565	39572							
V067	3641 #	9853	9860	9867	9874	9881	9888	9895	9901	9908	9915	9922
	9929	9936	9943	9950	9956	9963	9970	9977	9984	9991	9998	10005
	10011	10018	10025	10032	10039	10046	10053	10060	10066	10073	10080	10087
	10094	10101	10108	10115	10121	10128	10135	10142	10149	10156	10163	10170
	10176	10183	10190	10197	10204	10211	10218	10225	10231	10238	10245	10252
	10259	10266	10273	10280	10286	10293	10300	10307	10314	10321	10328	10335
	10341	10348	10355	10362	10369	10376	10383	10390	10396	10403	10410	10417
	10424	10431	10438	10445	10451	10458	10465	10472	10479	10486	10493	10500
	10506	10513	17649	17656	17663	17670	17677	17684	17695	17701	17708	17715
	17722	17729	17736	17743	17750	17756	17763	17770	17777	17784	17791	17798

Cross Reference Listing - Expression Names

	17805	17811	17818	17825	17832	17839	17846	17853	17860	17866	17873	17884
	17891	17898	17905	17912	17918	17925	17932	17943	17950	17957	17964	17970
	17977	18781	18788	18795	18802	18809	18816	18823	19748	19755	19762	19769
	19776	19783	19790	19796	19803	19810	19817	19824	19831	19838	19845	19851
	19860	19869	19876	19883	19890	19897	19903	19910	19919	19928	19935	19942
	19949	19955	19962	19969	19976	19983	19990	19997	20004	20010	20017	20829
	20838	20847	22112	22119	22126	22133	22140	22147	22154	22160	22167	22174
	22181	22188	22195	22202	22209	22215	25416	25423	25430	25437	25444	25451
	25458	25464	25471	25478	25485	33255	33262	33269	33276	33283	33290	33297
	33303	33310	33317	33324	33331	33338	33345	33351	33358	34776	36370	36377
	36384	36391	36398	36405	36987	36997	37007	37017	39517	39524	39531	39538
	39545	39552	39559	39565	39572							
V068	3642 #	43273	43276	43279	43282	43285	43288	43291	43294	43297	43300	43303
	43306	43309	43312	43315	43318	43322	43325	43328	43331	43334	43337	43340
	43343	43346	43349	43352	43355	43358	43361	43364	43367	43370	43373	43377
	43380	43383	43386	43389	43392	43395	43398	43401	43404	43407	43410	43413
	43416	43419	43422	43425	43428	43432	43435	43438	43441	43444	43447	43450
	43453	43456	43459	43462	43465	43468	43471	43474	43477	43480	43483	43487
	43490	43493	43496	43499	43502	43505	43508	43511	43514	43517	43520	43523
	43526	43529	43532	43535	43538	43542	43545	43548	43551	43554	43557	43560
	43563	43566	43569	43572	43575	43578	43581	43584	43587	43590	43593	43597
	43600	43603	43606	43609	43612	43615	43618	43621	43624	43627	43630	43633
	43636	43639	43642	43645	43648	43652	43655	43658	43661	43664	43667	43670
	43673	43676	43679	43682	43685	43688	43691	43694	43697	43700	43703	43707
	43710	43713	43716	43719	43722	43725	43728	43731	43734	43737	43740	43743
	43746	43749	43752	43755	43758	43762	43765	43768	43771	43774	43777	43780
	43783	43786	43789	43792	43795	43798	43801	43804	43807	43810	43813	43817
	43820	43823	43826	43829	43832	43835	43838	43841	43844	43847	43850	43853
	43856	43859	43862	43865	43868	43872	43875	43878	43881	43884	43887	43890
	43893	43896	43899	43902	43905	43908	43911	43914	43917	43920	43923	43927
	43930	43933	43936	43939	43942	43945	43948	43951	43954	43957	43960	43963
	43966	43969	43972	43975	43978	43982	43985	43988	43991	43994	43997	44000
	44003	44006	44009	44012	44015	44018	44021	44024	44027	44030	44033	44037
	44040	44043	44046	44049	44052							
V069	3643 #	43273	43276	43279	43282	43285	43288	43291	43294	43297	43300	43303
	43306	43309	43312	43315	43318	43322	43325	43328	43331	43334	43337	43340
	43343	43346	43349	43352	43355	43358	43361	43364	43367	43370	43373	43377
	43380	43383	43386	43389	43392	43395	43398	43401	43404	43407	43410	43413
	43416	43419	43422	43425	43428	43432	43435	43438	43441	43444	43447	43450
	43453	43456	43459	43462	43465	43468	43471	43474	43477	43480	43483	43487
	43490	43493	43496	43499	43502	43505	43508	43511	43514	43517	43520	43523
	43526	43529	43532	43535	43538	43542	43545	43548	43551	43554	43557	43560
	43563	43566	43569	43572	43575	43578	43581	43584	43587	43590	43593	43597
	43600	43603	43606	43609	43612	43615	43618	43621	43624	43627	43630	43633
	43636	43639	43642	43645	43648	43652	43655	43658	43661	43664	43667	43670
	43673	43676	43679	43682	43685	43688	43691	43694	43697	43700	43703	43707
	43710	43713	43716	43719	43722	43725	43728	43731	43734	43737	43740	43743
	43746	43749	43752	43755	43758	43762	43765	43768	43771	43774	43777	43780
	43783	43786	43789	43792	43795	43798	43801	43804	43807	43810	43813	43817
	43820	43823	43826	43829	43832	43835	43838	43841	43844	43847	43850	43853
	43856	43859	43862	43865	43868	43872	43875	43878	43881	43884	43887	43890
	43893	43896	43899	43902	43905	43908	43911	43914	43917	43920	43923	43927
	43930	43933	43936	43939	43942	43945	43948	43951	43954	43957	43960	43963
	43966	43969	43972	43975	43978	43982	43985	43988	43991	43994	43997	44000
	44003	44006	44009	44012	44015	44018	44021	44024	44027	44030	44033	44037

V07.57
V070

44040	44043	44046	44049	44052							
3576 #	39722	39736									
3646 #	5468	5468	5468	5547	5547	5553	5564	5569	5579	5596	5622
5630	5672	5672	5672	5687	5733	5754	5791	5800	5818	6139	6164
6167	6437	6499	6513	6517	6581	6592	6600	6608	6617	6621	6655
6657	6702	6769	6779	6779	6783	6787	6787	6791	6796	6796	6814
6818	6843	6900	6904	6907	6911	6914	6918	6922	7033	7033	7033
7033	7199	7199	7199	7220	7227	7318	7326	7334	7342	7349	7357
7365	7386	7398	7401	7413	7426	7429	7441	7445	7450	7462	7472
7485	7489	7495	7507	7511	7517	7529	7539	7796	7815	7925	7968
8052	8089	8147	8195	8240	8245	8261	8266	8278	8292	8292	8297
8301	8301	8307	8307	8312	8316	8316	8321	8321	8333	8351	8351
8351	8357	8357	8357	8363	8363	8368	8368	8373	8373	8378	8378
8388	8405	8405	8416	8416	8420	8420	8436	8436	8436	8441	8441
8441	8455	8455	8460	8460	8467	8467	8472	8472	8498	8501	8511
8522	8522	8522	8528	8536	8536	8553	8558	8564	8590	8590	8596
8596	8596	8603	8603	8606	8620	8620	8620	8636	8636	8645	8653
8653	8679	8679	8685	8685	8685	8692	8692	8695	8709	8709	8709
8727	8732	8737	8763	8763	8770	8770	8770	8777	8777	8780	8807
8807	8814	8814	8814	8821	8821	8824	8851	8851	8858	8858	8858
8866	8866	8869	8899	8899	8905	8905	8916	8916	3916	8932	8932
8932	8942	8942	8942	8946	8982	8986	8986	8986	8992	8996	9024
9045	9045	9050	9050	9059	9067	9072	9100	9149	9154	9159	9164
9170	9175	9180	9185	9191	9195	9195	9199	9203	9209	9214	9219
9224	9230	9234	9238	9242	9271	9271	9276	9276	9281	9281	9287
9287	9292	9292	9297	9297	9301	9306	9319	9319	9319	9329	9329
9329	9336	9336	9336	9341	9341	9341	9383	9502	9502	9506	9506
9515	9519	9523	9527	9554	9554	9565	9565	9581	9581	9591	9596
9600	9608	9614	9614	9619	9619	9625	9630	9634	9642	9648	9648
9653	9653	9687	9687	9707	9721	9721	9727	9727	9731	9731	9737
9737	9743	9743	9750	9750	9756	9756	9765	9765	9775	9775	9782
9782	9835	9835	9840	9840	10694	10694	10699	10699	10704	10704	10754
10754	10767	10767	10777	10777	10822	10822	10831	10831	11012	11016	11021
11116	11132	11136	11141	11166	11170	11177	11181	11226	11262	11262	11322
11387	11448	11459	11482	11482	11482	11489	11489	11489	11564	11672	11672
11672	11678	11678	11678	11691	11691	11691	11699	11699	11699	11706	11706
11706	11777	11777	11790	11790	11804	11808	11817	11899	11899	11910	11910
11917	11917	11930	11935	12607	12852	13081	13169	13169	13248	13248	13314
13325	13357	13367	13674	13774	13774	13800	13800	13951	13960	13970	13977
14209	14220	14235	14255	14317	14317	14343	14343	14462	14480	14646	14652
14673	14723	14723	14751	14751	14769	14774	14784	14804	14809	14816	14816
14822	14822	14949	14956	15163	15322	15353	15362	15367	15372	15376	15407
15407	15407	15478	15489	15496	15505	15511	15520	15527	15532	15538	15544
15695	15719	15731	15760	15783	15801	15852	15894	15911	15919	15988	16015
16021	16030	16035	16074	16214	16219	16270	16287	16488	16497	16512	16534
16539	16551	16570	16595	16611	16618	16638	16643	16742	16742	16742	16755
16755	16755	16771	16771	16771	16782	16812	16812	16812	16827	16827	16827
16852	16852	16868	16877	16894	16933	16950	16950	16957	16957	16997	17010
17014	17020	17034	17041	17056	17072	17081	17091	17102	17246	17261	17266
17286	17306	17457	17478	17494	17500	17500	17500	17536	17557	17589	17602
17626	17640	18199	18218	18236	18241	18245	18250	18275	18279	18279	18287
18291	18291	18316	18321	18331	18402	18711	18723	18734	18915	18971	18971
18971	18976	18976	18976	18981	18981	18981	18986	18986	18986	18991	18991
18991	18996	18996	18996	19001	19001	19001	19005	19005	19005	19010	19077
19093	19093	19098	19098	19098	19109	19113	19117	19121	19125	19129	19184

Cross Reference Listing - Expression Names

19184	19189	19189	19189	19250	19250	19250	19254	19254	19284	19284	19284
19293	19293	19299	19299	19330	19330	19330	19337	19343	19343	19382	19382
19382	19391	19391	19395	19444	19444	19444	19451	19451	19456	19519	19525
19540	19546	19559	19566	19572	19578	19590	19595	19640	19640	19645	19659
19659	19659	19664	19664	19664	19669	19669	19669	19674	19674	19674	19679
19679	19679	19684	19684	19684	19689	19689	19689	19693	19693	19693	19698
19723	19723	19728	19739	19739	20112	20118	20147	20156	20171	20192	20192
20195	20197	20208	20221	20230	20235	20240	20249	20258	20266	20275	20284
20293	20301	20310	20319	20328	20336	20345	20419	20425	20434	20441	20445
20455	20469	20475	20481	20481	20504	20509	20514	20531	20536	20553	20570
20581	20586	20591	20614	20623	20633	20643	20653	20663	20672	20682	20692
20702	20712	20721	20735	20739	20761	20765	20769	20816	20820	21026	21037
21037	21041	21041	21108	21108	21128	21133	21147	21147	21154	21154	21160
21160	21210	21218	21245	21250	21263	21272	21272	21362	21446	21450	21458
21464	21477	21483	21527	21532	21532	21539	21539	21550	21554	21559	21594
21612	21618	21625	21659	21675	21690	21705	21712	21726	21734	21758	21780
21798	21805	21814	21848	21861	21878	21885	21900	21907	21920	21965	21999
22004	22017	22050	22058	22093	22419	22463	22468	22478	22498	22516	22527
22537	22557	22576	22597	22615	22633	22654	22663	22684	22688	22699	22712
22717	22752	22761	22765	22781	23018	23018	23022	23022	23046	23086	23091
23091	23103	23145	23145	23155	23155	23159	23159	23183	23191	23191	23198
23201	23254	23258	23296	23296	23307	23311	23320	23320	23326	23329	23450
23450	23450	23454	23454	23454	23465	23465	23469	23469	23475	23521	23681
23704	23726	23731	23731	23731	23735	23735	23735	23757	23768	23768	23783
23783	23787	23787	23787	23804	23812	23821	23826	23829	23835	23839	23845
23845	23849	23849	23933	23980	23980	23980	23980	23985	23985	23985	23989
23989	23989	23989	24019	24027	24041	24041	24041	24041	24048	24048	24048
24074	24074	24074	24083	24083	24083	24083	24107	24115	24130	24130	24130
24130	24136	24136	24136	24191	24196	24196	24196	24202	24206	24216	24221
24228	24233	24241	24259	24265	24314	24314	24314	24324	24343	24343	24343
24377	24447	24457	24481	24561	24641	24644	24675	24693	24722	24747	24754
24770	24785	24793	24816	24825	24877	24938	24949	24970	24974	24984	24998
25002	25031	25037	25061	25071	25086	25095	25099	25104	25162	25173	25196
25210	25215	25263	25276	25289	25314	25319	25326	25383	25402	25602	25714
25921	25927	26090	26101	26212	26303	26303	26303	26345	26419	26425	26441
26608	26764	26815	26896	27197	27207	27234	27275	27298	27304	27494	27503
27506	27520	27674	27711	27728	27737	27754	27754	27754	27770	27770	27820
27890	27938	27942	27946	27965	27971	27980	27980	27980	27989	27992	27996
28004	28004	28004	28014	28018	28022	28061	28061	28079	28151	28172	28218
28241	28298	28402	28411	28490	28594	28619	28623	28633	28687	28687	28687
28691	28691	28691	28703	28810	28824	28828	28832	28836	28846	28846	28849
28866	28942	28946	28959	28962	28971	28979	28983	29093	29097	29100	29103
29109	29121	29121	29130	29146	29156	29173	29181	29185	29209	29224	29228
29235	29254	29258	29276	29281	29285	29334	29339	29346	29350	29370	29392
29398	29421	29424	29428	29556	29567	29571	29575	29620	29640	29694	29703
29707	29736	29749	29810	29843	29856	29903	29912	29931	29938	30002	30046
30085	30101	30118	30127	30132	30176	30190	30206	30206	30206	30215	30222
30267	30273	30286	30290	30326	30338	30346	30356	30360	30365	30374	30380
30388	30429	30440	30451	30485	30485	30485	30532	30536	30581	30581	30581
30611	30620	30645	30645	30645	30665	30665	30665	30670	30670	30670	30674
30674	30674	30710	30725	30725	30725	30747	30853	30857	30861	30865	30898
30898	30898	30904	30904	30904	30924	30938	30938	30938	30942	30942	30942
30948	30948	30948	30959	30963	30968	30974	30978	30984	30984	30984	30992
30992	30992	31034	31034	31034	31074	31166	31169	31175	31178	31214	31221
31238	31249	31412	31423	31432	31436	31535	31539	31549	31549	31575	31575

Cross Reference Listing - Expression Names

31579	31675	31675	31692	31745	31857	31860	31865	31871	31881	31893	31915
31920	31933	31948	32034	32034	32037	32041	32085	32112	32115	32119	32140
32143	32152	32155	32159	32408	32412	32435	32442	32446	32456	32463	32467
32475	32485	32491	32498	32502	32692	32692	32844	32848	32889	32894	32907
32926	32930	32965	32965	32970	32970	33002	33006	33011	33031	33041	33120
33131	33134	33147	33147	33147	33156	33164	33167	33172	33176	33245	33467
33475	33500	33503	33515	33572	33572	33572	33624	33624	33624	33716	33722
33728	33732	33739	33783	33836	33953	33994	34008	34012	34038	34187	34287
34298	34305	34310	34369	34380	34437	34447	34451	34463	34495	34501	34510
34607	34613	34620	34624	34654	34663	34670	34676	34710	34720	34737	34741
34745	35027	35257	35276	35309	35321	35334	35349	35360	35363	35441	35446
35451	35547	35580	35629	35632	35655	35951	35972	35978	35983	36015	36021
36035	36057	36065	36073	36125	36147	36161	36292	36300	36342	36357	36361
36466	36470	36488	36518	36523	36550	36554	36572	36636	36639	36777	36813
36818	36871	36878	36882	36892	37144	37331	37346	37353	37357	37361	37365
37369	37479	37492	37679	37981	37990	37994	37998	38002	38006	38011	38040
38169	38367	38419	38453	38483	38487	38541	38566	38572	38616	38622	38659
38664	38734	38776	38823	38827	38827	38834	38837	38837	38845	39003	39008
39016	39025	39049	39056	39063	39075	39099	39106	39147	39167	39185	39335
39335	39335	39338	39341	39341	39398	39508	39796	40028	40042	40201	40371
40383	40557	40568	40577	40586	40594	40622	40630	40657	40687	40715	40719
40751	40775	40778	40810	40854	40991	40991	40991	40996	41001	41001	41001
41007	41007	41007	41027	41027	41027	41032	41038	41038	41038	41045	41045
41045	41065	41065	41065	41070	41076	41076	41076	41083	41083	41083	41103
41103	41103	41108	41114	41114	41114	41121	41121	41121	41140	41145	41150
41156	41176	41176	41181	41186	41186	41193	41193	41206	41215	41215	41225
41233	41233	41252	41260	41260	41260	41265	41271	41271	41271	41276	41282
41282	41282	41306	41316	41323	41323	41329	41329	41333	41333	41342	41405
41414	41420	41425	41448	41448	41452	41486	41486	41490	41494	41494	41500
41505	41525	41525	41529	41533	41533	41539	41544	41568	41568	41568	41573
41573	41573	41591	41600	41600	41604	41608	41608	41614	41619	41628	41628
41628	41657	41657	41657	41728	41763	41763	41763	41772	41772	41772	41777
41782	41787	41802	41807	41807	41821	41821	41821	41826	41831	41831	41831
41837	41837	41854	41854	41860	41860	41860	41869	41886	41886	41892	41892
41892	41901	41917	41923	41932	41948	41948	42015	42015	42021	42021	42029
42029	42033	42033	42038	42038	42083	42083	42109	42150	42150	42159	42164
42211	42211	42211	42217	42222	42222	42222	42243	42243	42249	42249	42259
42276	42282	42292	42309	42309	42315	42315	42324	42357	42376	42384	42389
42393	42404	42404	42439	42448	42479	42538	42538	42542	42546	42546	42551
42559	42559	42566	42566	42571	42571	42581	42581	42604	42604	42604	42621
42621	42621	42626	42631	42631	42631	42652	42652	42655	42658	42716	42716
42720	42769	42769	42769	42774	42774	42779	42779	42779	42785	42785	42785
42794	42794	42794	42799	42799	42799	42799	42821	42831	42831	42831	42840
42847	42847	42847	42847	42860	42860	42866	42918	42918	42918	42922	42922
42928	42928	42928	42935	42935	42935	42944	42944	42944	42949	42949	42949
42949	42967	42977	42977	42977	42986	42993	42993	42993	42993	43007	43059
43059	43059	43063	43063	43069	43069	43069	43076	43076	43076	43084	43084
43084	43088	43088	43088	43088	43118	43118	43126	43132	43132	43132	43153
43153	43159	43169	43169	43169	43169	43199	43199	43199	43204	43204	43204
43204	43209	43209	43209	43216	43227	43232	43237	43243	43252	44358	44358
44369	44369	44374	44374	44374	44379	44379	44384	44384	44393	44393	44393
44398	44398	44416	44426	44426	44426	44438	44444	44456	44456	44456	44477
44477	44485	44485	44489	44489	44489	44493	44493	44497	44497	44505	44505
44505	44508	44508	44523	44529	44529	44534	44539	44545	44545	44545	44551
44572	44572	44580	44580	44585	44585	44585	44589	44589	44593	44593	44601

Cross Reference Listing - Expression Names

	44601	44601	44604	44604	44625	44625	44635	44641	44641	44641	44662	44662
	44670	44670	44675	44675	44675	44679	44679	44683	44683	44691	44691	44691
	44694	44694	44709	44714	44719	44719	44724	44730	44730	44730	44759	44759
	44763	44763	44763	44767	44767	44771	44771	44779	44779	44779	44782	44782
	44799	44805	44837	44837	44842	44842	44842	44847	44847	44852	44852	44861
	44861	44861	44866	44866	44884	44891	44891	44891	44898	44904	44915	44915
	44915	44951	44951	44951	44956	44956	44960	44960	44960	44970	44970	44970
	44974	44974	45002	45006	45006	45012	45012	45027	45033	45033	45040	45040
	45040	45058	45075	45075	45080	45080	45080	45085	45085	45089	45089	45089
	45098	45098	45098	45103	45103	45131	45138	45138	45138	45143	45143	45149
	45149	45187	45187	45194	45194	45194	45201	45201	45207	45207	45217	45217
	45217	45222	45222	45235	45240	45247	45247	45247	45292	45292	45297	45297
	45297	45302	45302	45306	45306	45315	45315	45315	45320	45320	45338	45365
	45365	45375	45375	45380	45380	45380	45385	45385	45390	45390	45399	45399
	45399	45404	45404	45420	45452	45452	45462	45462	45467	45467	45467	45472
	45472	45477	45477	45486	45486	45486	45491	45491	45511	45525	45525	45525
	45530	45530	45542	45548	45548	45554	45565	45565	45565	45599	45599	45608
	45608	45613	45613	45613	45618	45618	45623	45623	45632	45632	45632	45637
	45637	45654	45664	45678	45683	45683	45718	45718	45723	45723	45723	45728
	45728	45733	45733	45742	45742	45742	45747	45747	45767	45775	45775	45775
	45781	45790	45802	45802	45802	45818	45818	45818	45834	45834	45839	45839
	45839	45844	45844	45849	45849	45858	45858	45858	45863	45863	45881	45910
	45910	45922	45922	45928	45928	45928	45934	45934	45939	45939	45950	45950
	45950	45956	45956	45974	45987	45987	46025	46025	46030	46030	46030	46035
	46035	46040	46040	46049	46049	46049	46054	46054	46071	46079	46085	46171
	46176	46394	46419	46444	46768	46871	46879	46883	46883	46883	46890	46890
	46890	46915	46990	46990	46990	46996	47000	47000	47000	47009	47009	47009
	47015	47015	47015	47015	47019	47053	47059	47083	47144	47150	47197	47237
	47332	47346	47361	47366	47382	47398	47631	47641	47681	47691	47798	47815
	47868	47886	47889	47893	48029							
V071	3647 #	5328	5328	5335	5335	5358	5358	5366	5366	5385	5385	5399
	5399	5403	5403	5409	5409	5455	5455	5459	5459	5463	5463	5472
	5472	5477	5477	5488	5488	5496	5496	5500	5500	5504	5504	5508
	5508	5515	5515	5519	5519	5523	5523	5531	5531	5535	5535	5540
	5540	5560	5560	5583	5583	5591	5591	5601	5601	5605	5605	5614
	5614	5617	5617	5626	5626	5630	5630	5659	5659	5662	5662	5676
	5676	5682	5682	5687	5687	5699	5699	5712	5712	5716	5716	5723
	5723	5733	5733	5750	5750	5785	5785	5796	5796	5824	5824	5899
	5899	5913	5913	5920	5920	5934	5934	5940	5940	5943	5943	5951
	5951	5954	5954	5957	5957	5963	5963	5968	5968	5978	5978	5992
	5992	5998	5998	6007	6007	6128	6128	6132	6132	6142	6142	6149
	6149	6156	6156	6160	6160	6171	6171	6174	6174	6221	6221	6225
	6225	6229	6229	6233	6233	6236	6236	6240	6240	6244	6244	6248
	6248	6256	6256	6265	6265	6270	6270	6274	6274	6281	6281	6285
	6285	6289	6289	6293	6293	6299	6299	6303	6303	6307	6307	6313
	6313	6317	6317	6321	6321	6327	6327	6331	6331	6335	6335	6341
	6341	6345	6345	6349	6349	6355	6355	6359	6359	6363	6363	6369
	6369	6373	6373	6377	6377	6383	6383	6387	6387	6391	6391	6397
	6397	6401	6401	6405	6405	6411	6411	6415	6415	6419	6419	6425
	6425	6433	6433	6464	6464	6471	6471	6478	6478	6487	6487	6503
	6503	6508	6508	6528	6528	6536	6536	6542	6542	6545	6545	6557
	6557	6613	6613	6662	6662	6667	6667	6695	6709	6709	6753	6753
	6803	6803	6825	6825	6832	6832	6858	6858	6932	6932	6965	6965
	6978	6978	7014	7014	7033	7037	7037	7042	7042	7048	7048	7051
	7051	7055	7055	7059	7059	7062	7062	7066	7066	7071	7071	7074

Cross Reference Listing - Expression Names

7074	7078	7078	7083	7083	7086	7086	7090	7090	7095	7095	7098
7098	7102	7102	7107	7107	7110	7110	7166	7166	7170	7170	7173
7173	7177	7177	7184	7184	7187	7187	7231	7231	7235	7235	7264
7264	7268	7268	7276	7276	7280	7280	7284	7284	7310	7310	7314
7314	7322	7322	7330	7330	7338	7338	7345	7345	7353	7353	7361
7361	7369	7369	7373	7373	7378	7378	7383	7383	7405	7405	7410
7410	7420	7420	7433	7433	7438	7438	7454	7454	7459	7459	7466
7466	7476	7476	7482	7482	7499	7499	7504	7504	7521	7521	7526
7526	7533	7533	7543	7543	7548	7548	7557	7557	7561	7561	7565
7565	7569	7569	7578	7578	7581	7581	7591	7591	7595	7595	7615
7615	7627	7627	7635	7635	7644	7644	7648	7648	7652	7652	7664
7664	7676	7676	7690	7690	7706	7706	7714	7714	7718	7718	7729
7729	7741	7741	7746	7746	7750	7750	7753	7753	7757	7757	7762
7762	7769	7769	7789	7789	7809	7809	7829	7829	7833	7833	7850
7850	7861	7861	7873	7873	7908	7908	7912	7912	7915	7915	7944
7944	7949	7949	7958	7958	7963	7963	7975	7975	7986	7986	8009
8009	8014	8014	8026	8026	8032	8032	8058	8058	8063	8063	8068
8068	8072	8072	8076	8076	8080	8080	8094	8094	8143	8143	8187
8187	8191	8191	8199	8199	8206	8206	8910	8927	8950	9027	9039
9315	9324	9396	9396	9401	9401	9416	9420	9420	9425	9430	9430
9441	9449	9449	9459	9459	9559	9559	9570	9570	9670	9670	9690
9690	9694	9770	9770	9778	9778	9830	10817	10817	10825	10825	10868
10868	10879	10879	10926	10926	10930	10930	10934	10934	11029	11029	11034
11034	11149	11149	11154	11154	11329	11329	11337	11340	11340	11353	11499
11712	11771	11771	11782	11782	11799	11799	11813	11813	11885	11885	12035
12035	12042	12042	12224	12238	12238	12333	12333	12343	12343	12372	12372
12385	12385	12400	12400	12513	12513	12519	12519	12550	12550	12556	12562
12562	12580	12580	12594	12594	12621	12621	12633	12637	12637	12642	12642
12700	12700	12706	12706	12712	12712	12716	12716	12721	12721	12726	12726
12731	12731	12734	12734	12858	12858	12878	12878	13085	13085	13469	13480
13480	13588	13588	13625	13625	13670	13670	13684	13684	13702	13702	13878
13878	14088	14088	14113	14113	14118	14118	14426	14426	14442	14442	14446
14446	14501	14501	14505	14505	14666	14666	14690	14690	14702	14702	14713
14713	14735	14735	14837	14837	14843	14843	14973	14973	15008	15008	15016
15016	15073	15073	15079	15079	15084	15084	15088	15088	15095	15095	15098
15098	15123	15123	15166	15166	15330	15335	15386	15390	15390	15418	15418
15422	15422	15426	15426	15690	15690	15711	15711	15742	15742	15777	15777
15812	15812	15830	15830	15847	15847	15876	15876	15880	15880	15889	15905
15905	15997	15997	16060	16060	16064	16064	16069	16069	16114	16114	16120
16120	16137	16137	16141	16141	16200	16200	16230	16230	16290	16290	16435
16435	16483	16483	16525	16525	16563	16563	16574	16574	16599	16599	16604
16604	16624	16624	16668	16668	16991	16991	16997	17118	17118	17123	17123
17160	17160	17169	17169	17173	17173	17177	17177	17238	17238	17246	17474
17474	17562	17562	17566	17566	17592	17592	17610	17610	17634	17634	19042
19042	19053	19053	19065	19065	19077	19077	19154	19154	19167	19167	19237
19237	19242	19242	19304	19311	19316	19349	19354	19359	19400	19405	19470
19470	19479	19479	19484	19484	19583	19583	20124	20124	20129	20129	20135
20135	20140	20140	20156	20156	20156	20177	20177	20200	20200	20211	20211
20244	20244	20253	20253	20262	20262	20270	20270	20279	20279	20288	20288
20297	20297	20305	20305	20314	20314	20323	20323	20332	20332	20340	20340
20350	20350	20355	20355	20360	20360	20365	20365	20370	20370	20375	20375
20379	20379	20384	20384	20389	20389	20394	20394	20399	20399	20404	20404
20409	20409	20429	20429	20438	20438	20450	20450	20455	20455	20465	20465
20469	20469	20491	20491	20495	20495	20499	20499	20536	20536	20540	20540
20545	20545	20548	20548	20553	20553	20573	20573	20608	20608	20614	20614

Cross Reference Listing - Expression Names

20618	20618	20623	20623	20627	20627	20633	20633	20637	20637	20643	20643
20647	20647	20653	20653	20657	20657	20663	20663	20667	20667	20672	20672
20676	20676	20682	20682	20686	20686	20692	20692	20696	20696	20702	20702
20706	20706	20712	20712	20716	20716	20721	20721	20742	20742	20751	20751
20780	20780	20788	20788	20792	20792	20806	20806	20811	20811	20901	20901
20914	20914	20930	20930	20950	20955	20955	20962	20962	20966	20966	20972
20972	20976	20976	21032	21046	21051	21102	21102	21117	21117	21137	21137
21144	21166	21166	21206	21206	21213	21213	21263	21263	21320	21320	21324
21324	21327	21327	21350	21386	21386	21396	21396	21412	21439	21439	21454
21454	21468	21468	21472	21472	21543	21543	21546	21546	21604	21604	21615
21615	21621	21621	21655	21655	21665	21665	21698	21698	21730	21730	21758
21758	21787	21787	21794	21794	21808	21808	21848	21848	21858	21858	21868
21868	21874	21874	21889	21889	21903	21903	21912	21912	21952	21952	21969
21969	21979	21979	21999	21999	22004	22004	22012	22012	22021	22021	22429
22429	22482	22482	22486	22486	22547	22547	22638	22638	22677	22677	22785
22785	23001	23001	23004	23004	23031	23031	23035	23035	23039	23039	23050
23050	23079	23079	23096	23096	23107	23107	23129	23129	23168	23168	23172
23172	23176	23176	23187	23187	23205	23205	23237	23237	23245	23245	23279
23279	23333	23333	23437	23437	23469	23479	23688	23688	23694	23761	23761
23808	23808	23818	23818	23922	23922	23947	23947	23957	23957	24015	24015
24062	24062	24103	24103	24251	24251	24314	24343	24367	24367	24423	24423
24463	24463	24472	24472	24477	24477	24486	24495	24495	24526	24526	24541
24541	24647	24647	24665	24665	24669	24669	24680	24680	24684	24684	24708
24708	24760	24760	24782	24782	24797	24797	24802	24802	24805	24805	24835
24835	24869	24869	24930	24930	24933	24933	24952	24952	24959	24959	25025
25025	25041	25041	25047	25047	25051	25051	25092	25092	25110	25110	25119
25119	25165	25165	25204	25204	25243	25243	25284	25284	25292	25292	25299
25299	25368	25368	25612	25612	25634	25634	25650	25650	25677	25677	25681
25681	25714	25714	25792	25792	25797	25797	25973	25973	25994	25994	26003
26003	26006	26006	26009	26009	26017	26017	26029	26029	26111	26111	26143
26143	26146	26146	26172	26212	26212	26293	26293	26361	26361	26364	26364
26370	26370	26373	26373	26465	26465	26501	26501	26504	26504	26583	26583
26790	26790	26794	26794	26848	26848	26853	26853	26896	26896	27014	27014
27017	27017	27023	27023	27026	27026	27044	27044	27051	27051	27063	27063
27067	27067	27102	27102	27333	27333	27375	27375	27379	27379	27665	27665
27687	27687	27732	27732	27775	27775	27779	27779	27797	27797	27801	27801
27817	27817	27827	27827	27835	27835	27839	27839	27845	27845	27849	27849
27855	27855	27859	27859	27868	27868	27872	27872	27876	27876	27911	27911
27916	27916	27984	27984	28008	28008	28027	28027	28049	28049	28065	28065
28075	28075	28106	28106	28117	28117	28144	28144	28147	28147	28155	28155
28178	28178	28183	28183	28188	28188	28213	28213	28245	28245	28249	28264
28264	28268	28268	28279	28285	28285	28298	28298	28312	28312	28324	28328
28328	28334	28334	28338	28338	28341	28341	28418	28418	28428	28428	28453
28453	28477	28477	28494	28494	28500	28500	28506	28506	28534	28534	28538
28538	28563	28563	28584	28589	28598	28598	28601	28601	28619	28619	28661
28661	28665	28665	28670	28670	28770	28770	28774	28774	28779	28779	28783
28783	28793	28793	28797	28797	28824	28824	28828	28828	28836	28836	28855
28855	29074	29074	29089	29089	29112	29112	29116	29116	29134	29134	29138
29138	29142	29142	29176	29176	29246	29246	29250	29250	29269	29269	29381
29381	29488	29488	29491	29491	29511	29511	29535	29535	29538	29538	29560
29560	29567	29567	29585	29585	29590	29590	29595	29595	29600	29600	29605
29605	29636	29636	29660	29660	29670	29670	29680	29680	29684	29684	29689
29698	29698	29712	29712	29717	29717	29723	29723	29727	29727	29731	29818
29818	29822	29822	29828	29828	29836	29836	29843	29843	29850	29850	29869
29869	29877	29877	29912	29912	29986	30029	30105	30105	30202	30202	30226

Cross Reference Listing - Expression Names

30226	30244	30244	30249	30249	30253	30253	30257	30257	30267	30267	30273
30273	30277	30277	30281	30281	30286	30286	30296	30296	30352	30352	30374
30374	30380	30380	30388	30388	30451	30451	30457	30457	30461	30461	30481
30481	30492	30492	30499	30499	30518	30518	30532	30532	30575	30581	30586
30586	30601	30601	30611	30611	30620	30620	30626	30626	30632	30632	30636
30636	30641	30641	30714	30714	30735	30735	30756	30756	30760	30760	30781
30781	30787	30787	30830	30875	30875	30879	30879	30888	30888	30909	30909
30926	30926	30987	30987	30998	30998	31002	31002	31009	31009	31014	31014
31021	31021	31026	31026	31049	31049	31062	31062	31112	31112	31227	31227
31230	31230	31241	31241	31349	31349	31442	31442	31456	31456	31495	31498
31498	31505	31505	31514	31514	31588	31588	31650	31650	31688	31688	31704
31704	31708	31708	31745	31745	31926	31926	32044	32044	32082	32082	32100
32100	32124	32124	32131	32131	32163	32163	32172	32172	32372	32372	32379
32379	32385	32385	32396	32396	32400	32400	32404	32404	32420	32420	32423
32423	32426	32426	32438	32438	32459	32459	32478	32478	32482	32482	32494
32494	32511	32511	32545	32545	32564	32564	32567	32567	32570	32570	32574
32574	32591	32591	32595	32595	32599	32599	32603	32603	32607	32607	32611
32611	32615	32615	32625	32625	32629	32629	32635	32635	32639	32639	32645
32645	32649	32649	32655	32655	32659	32659	32664	32664	32667	32667	32671
32671	32687	32687	32692	32702	32702	32707	32707	32747	32747	32791	32791
32815	32815	32823	32823	32829	32829	32833	32833	32837	32837	32862	32862
32880	32880	32884	32884	32903	32903	32916	32916	32920	32920	32938	32938
32952	32952	32984	32984	32992	32992	32996	32996	33019	33019	33027	33027
33037	33037	33051	33051	33061	33061	33065	33065	33128	33128	33150	33150
33467	33467	33484	33484	33527	33527	33535	33535	33538	33538	33542	33542
33545	33545	33559	33575	33575	33579	33579	33611	33611	33615	33615	33619
33619	33629	33629	33639	33639	33650	33650	33650	33654	33654	33660	33660
33669	33669	33673	33673	33677	33677	33820	33820	33831	33831	33904	33904
33917	33917	33956	33956	33999	33999	34005	34005	34029	34029	34033	34033
34072	34072	34076	34076	34089	34089	34093	34093	34133	34133	34147	34147
34150	34150	34154	34154	34157	34157	34168	34168	34172	34172	34178	34178
34198	34198	34241	34241	34260	34260	34264	34264	34268	34268	34275	34275
34294	34294	34301	34301	34319	34319	34351	34363	34363	34380	34380	34394
34394	34405	34405	34441	34441	34458	34458	34498	34498	34505	34505	34561
34561	34570	34570	34610	34610	34658	34658	34679	34679	34682	34682	34701
34701	34704	34707	34707	34714	34714	34838	34838	34864	34864	34891	34891
34900	34900	34904	34904	34908	34908	34913	34913	35014	35014	35031	35031
35064	35064	35067	35067	35079	35079	35083	35083	35091	35091	35107	35107
35111	35111	35115	35115	35119	35119	35123	35123	35127	35127	35176	35176
35186	35186	35205	35205	35252	35252	35268	35268	35272	35288	35288	35315
35315	35328	35328	35331	35331	35363	35369	35369	35372	35372	35376	35376
35382	35382	35385	35385	35390	35390	35394	35394	35402	35402	35405	35405
35408	35408	35412	35412	35418	35418	35437	35437	35456	35456	35460	35460
35469	35469	35478	35478	35482	35482	35489	35489	35493	35493	35498	35498
35503	35503	35512	35512	35516	35516	35522	35522	35526	35526	35532	35532
35536	35536	35567	35567	35571	35571	35576	35576	35585	35585	35590	35590
35599	35599	35640	35651	35660	35670	35670	35698	35698	35703	35703	35741
35741	35779	35779	35779	35783	35783	35798	35798	35802	35802	35814	35814
35818	35818	35822	35822	35826	35826	35834	35834	35838	35838	35842	35842
35850	35850	35853	35853	35857	35857	35861	35861	35873	35873	35897	35897
35905	35905	35914	35914	35919	35919	35923	35923	35927	35927	35931	35931
35935	35935	35939	35939	35987	35987	36000	36000	36004	36004	36007	36007
36047	36047	36053	36053	36060	36060	36063	36063	36083	36083	36088	36088
36092	36092	36097	36097	36101	36101	36106	36106	36112	36112	36116	36116
36121	36121	36136	36136	36139	36139	36152	36152	36156	36156	36184	36184

Cross Reference Listing - Expression Names

```
                46648   46648   46655   46655   46662   46662   46668   46668   46675   46675   46682   46682
                46689   46689   46696   46696   46716   46716   46725   46725   46731   46731   46740   46740
                46744   46744   46749   46749   46763   46763   46781   46781   46807   46807   46810   46810
                46815   46815   46819   46819   46827   46827   46832   46832   46875   46875   46886   46886
                46893   46893   46895   46895   46908   46911   46911   46922   46922   46926   46926   46947
                46947   46952   46952   46990   46993   46993   47000   47003   47003   47012   47012   47022
                47022   47062   47062   47070   47070   47091   47091   47094   47094   47098   47098   47101
                47101   47109   47109   47112   47112   47136   47136   47139   47139   47154   47154   47158
                47158   47162   47162   47165   47165   47169   47169   47173   47173   47176   47176   47181
                47181   47187   47187   47190   47190   47197   47197   47201   47201   47208   47208   47211
                47211   47215   47215   47219   47219   47222   47222   47227   47227   47230   47230   47237
                47237   47241   47241   47248   47248   47251   47251   47255   47255   47259   47259   47262
                47262   47265   47265   47271   47271   47281   47281   47295   47295   47299   47299   47302
                47302   47338   47338   47338   47356   47356   47371   47371   47371   47378   47378   47393
                47393   47402   47402   47406   47406   47412   47412   47415   47415   47423   47423   47431
                47431   47470   47470   47474   47474   47478   47478   47495   47495   47500   47500   47505
                47505   47510   47510   47515   47515   47520   47520   47541   47541   47546   47546   47551
                47551   47561   47561   47566   47570   47575   47575   47647   47647   47651   47651   47655
                47655   47661   47661   47740   47740   47752   47759   47759   47782   47782   47805   47805
                47810   47810   47899   47899   47904   47904   47937   47937   47943   47943   47947   47947
                47984   47984   48033   48033   48084   48084   48087   48087   48090   48090   48093   48093
                48096   48096   48099   48099
V072            3648 #  6483    6800    7113    7639    7655    7667    7686    7782    7786    7792    7812
                7905    7900    7920    11383   12603   17531   17547   18402   27499   27513   27526   27534
                28559   28574   28725   28746   30747   30826   30915   30919   32200   32203   32206   32211
                32237   32240   32243   32248   32274   32277   32280   32285   33208   33563   35433   35954
                35958   36043   36069   36128   38953   38971   38975   40300   40484   40489   40494   40506
                40532   40538   40622   40630   45058
V080            3651 #  5381    5492    6135    6145    6695    7132    7955    8020    8036    8041    8046
                8982    8989    9112    9121    9401    9486    12633   14717   14745   14859   14875   15386
                16742   16755   16771   16782   16812   16827   16852   16868   16877   16883   16894   16917
                16933   16939   16950   16957   16980   17102   17254   17299   20156   20801   20950   21315
                21350   21412   21984   22025   22050   22099   23744   23748   23752   23792   23797   23858
                23862   23866   24661   25173   25725   26871   26880   28959   29209   29765   29846   30029
                31169   31309   31740   31748   31865   33245   33559   34704   35051   35055   35387   35543
                35651   35660   35779   35783   35802   35810   35897   35948   36010   36083   36088   36097
                36121   36248   36345   36350   36619   36796   36896   37182   37189   37208   37213   37219
                37336   37690   37709   37804   37827   37841   37864   37918   38035   38084   38098   38149
                38206   38219   38362   38449   38529   38534   38556   38763   38817   38894   38898   38906
                38924   38962   39034   39154   39944   39955   40858   41362   41385   41561   41643   41664
                41693   41698   41703   42041   42171   42508   42670   42683   47278   47570
V09.21          3577 #  38146
V10.35          3578 #  35698
V11.34          3579 #  35585   35599   35612   35625   35703   36679   37810   37816   40250   40365   40441
                40657   47193   47204   47233   47244
V20.70          3580 #  5564    5569    5579    5596    5630    5672    6900    6907    6914    7227    8292
                8351    8357    8363    8373    8405    8416    8420    8436    8441    8460    8472    8501
                8522    8528    8536    8558    8596    8606    8620    8653    8685    8695    8709    8732
                8770    8780    8814    8824    8858    8869    8916    8932    8942    8986    9045    9050
                9072    9319    9329    9336    9341    9502    9515    9519    9523    9527    9687    9721
                9727    9731    9737    9743    9750    9756    9765    9775    9782    9835    9840    10694
                10699   10704   10754   10767   10777   10822   10831   11262   11482   11489   11672   11678
                11691   11699   11706   11777   11790   11899   11910   11917   11930   11935   13169   13248
                13774   13800   14317   14343   14723   14751   14769   14774   14804   14809   14816   14822
                15322   15489   15496   16742   16755   16771   16782   16812   16827   16868   16877   16894
```

	16933	16950	16957	18241	18245	18279	18291	18711	18723	18734	19109	19113
	19117	19121	19125	19129	19293	19299	19391	19451	19645	19728	20197	20230
	20240	20249	20258	20266	20275	20284	20293	20301	20310	20319	20328	20336
	20345	20425	2043'	20441	20455	20469	21037	21041	21147	21362	21458	21464
	21483	21539	21550	21612	21618	21625	21705	21712	21726	21734	21805	21814
	21900	21907	21920	21999	22004	22017	22058	22093	23046	23103	23183	23258
	23307	23311	23475	23521	25714	26212	26896	28172	28298	28594	28619	28623
	28633	28810	28824	28828	28832	28836	29694	29703	29707	30002	30046	30127
	30132	30267	30273	30286	30290	30356	30374	30380	30388	30532	30536	30611
	30620	30710	30747	30853	30857	30861	30865	30924	30959	30963	30974	30978
	31745	32844	32848	32926	32930	32965	32970	33011	33031	33041	33953	34437
	34495	36361	36777	36813	37353	37361	37369	37994	37998	38002	[illegible]006	38011
	38040	38483	38487	38572	38664	40991	41001	41007	41027	41038	41045	41065
	41076	41083	41103	41114	41121	41193	41215	41260	41271	41282	41500	41539
	41568	41573	41628	41657	41763	41772	41821	41831	41837	41854	41860	41886
	41892	42211	42222	42243	42309	42376	42384	42393	42538	42546	42551	42566
	42571	42621	42631	46171	46176	46871	46879	47053	47059	47144	47332	47361
	47382											
V21.50-54.70	3583 #	9170	9175	9180	9185	9191	9195	9199	9203	9209	9214	9219
	9224	9230	9234	9238	9242	9554	9565	9581	9591	9596	9600	9608
	9614	9619	9625	9630	9634	9642	9648	9653	12852	13081	14209	14220
	14235	14255	14673	15783								
V22.70	3581 #	5547	5622	5687	5733	5754	5791	5800	5818	6779	6787	6796
	8645	15719	15760	16021	16035	16219	16611	16618	16643	17010	17014	17020
	17041	17081	17091	17261	17266	17286	17557	19077	19540	19546	19739	20112
	20118	20536	20581	20586	20591	20614	20623	20633	20643	20653	20663	20672
	20682	20692	20702	20712	20721	20761	21272	21446	21527	21532	21659	21780
	21861	21878	21965	22050	23757	23804	24202	24206	24481	26345	29398	29843
	29912	30118	30215	30451	31074	32408	32412	32435	32456	32475	32491	33500
	33515	34380	36871	36882	39335	39338	42604	42652	42716	42821	42831	42847
	42967	42977	42993	43169	43216	43243	43252	44416	44426	44438	44444	44456
	44539	44545	44635	44641	44724	44730	44884	44891	44898	44904	44915	45027
	45040	45131	45138	45240	45247	45338	45420	45511	45525	45542	45548	45554
	45565	45654	45664	45767	45775	45781	45790	45802	45881	45974	46071	46079
	46085	46768	47150	47197	47237	47346	47366					
V24.25.70	3582 #	20156	20553	21108	21210	21218	21594	21675	21690	21798	21885	29209
	29620	35632	36125	36470	36488	36554	36572	38419	39796	40028	40042	40201
	40371	40383	40557	40568	40577	40586	40594	40622	40630	40657	40687	40751
	40810	40854										
V50.51.70	3584 #											
V71.80	3585 #	37213	37219	38529	38556							
VIELD.R1H	2378 #	18205	18224	18359	18667							
VIELD.R1L	2377 #	18205	18224	18359	18667							
VIELD.R2H	2380 #	18205	18224	18359	18667							
VIELD.R2L	2379 #	18205	18224	18359	18667							
VIELD.R3H	2382 #	18205	18224	18359	18667							
VIELD.R3L	2381 #	18205	18224	18359	18667							
WBUS.DRIVE	3652 #	5381	5492	6135	6145	6695	7132	7955	8020	8036	8041	8046
	8982	8989	9112	9121	9401	9486	12633	14717	14745	14859	14875	15386
	16742	16755	16771	16782	16812	16827	16852	16868	16877	16883	16894	16917
	16933	16939	16950	16957	16980	17102	17254	17299	20156	20801	20950	21315
	21350	21412	21984	22025	22050	22099	23744	23748	23752	23792	23797	23858
	23862	23866	24661	25173	25725	26871	26880	28959	29209	29765	29846	30029
	31169	31309	31740	31748	31865	33245	33559	34704	35051	35055	35387	35543
	35651	35660	35779	35783	35802	35810	35897	35948	36010	36083	36088	36097

Cross Reference Listing - Expression Names

	36121	36248	36345	36350	36619	36796	36896	37182	37189	37208	37213	37219
	37336	37690	37709	37804	37827	37841	37864	37918	38035	38084	38098	38149
	38206	38219	38362	38449	38529	38534	38556	38763	38817	38894	38898	38906
	38924	38962	39034	39154	39944	39955	40858	41362	41385	41561	41643	41664
	41693	41698	41703	42041	42171	42508	42670	42683	47278	47570		
WCS.R1H	2517 #											
WCS.R1L	2516 #											
WCS.R2H	2519 #											
WCS.R2L	2518 #											
WCS.R3H	2521 #											
WCS.R3L	2520 #											
WCTRL.07	3665 #	5381	5492	6135	6145	6695	7132	7955	8020	8036	8041	8046
	8982	8989	9112	9121	9401	9486	12633	14717	14745	14859	14875	15386
	16742	16755	16771	16782	16812	16827	16852	16868	16877	16883	16894	16917
	16933	16939	16950	16957	16980	17102	17254	17299	20156	20801	20950	21315
	21350	21412	21984	22025	22050	22099	23744	23748	23752	23792	23797	23858
	23862	23866	24661	25173	25725	26871	26880	28959	29209	29765	29846	30029
	31169	31309	31740	31748	31865	33245	33559	34704	35051	35055	35387	35543
	35651	35660	35779	35783	35802	35810	35897	35948	36010	36083	36088	36097
	36121	36248	36345	36350	36619	36796	36896	37182	37189	37208	37213	37219
	37336	37690	37709	37804	37827	37841	37864	37918	38035	38084	38098	38149
	38206	38219	38362	38449	38529	38534	38556	38763	38817	38894	38898	38906
	38924	38962	39034	39154	39944	39955	40858	41362	41385	41561	41643	41664
	41693	41698	41703	42041	42171	42508	42670	42683	47278	47570		
WCTRL.GROUP	3664 #	5381	5492	6135	6145	6695	7132	7955	8020	8036	8041	8046
	8982	8989	9112	9121	9401	9486	12633	14717	14745	14859	14875	15386
	16742	16755	16771	16782	16812	16827	16852	16868	16877	16883	16894	16917
	16933	16939	16950	16957	16980	17102	17254	17299	20156	20801	20950	21315
	21350	21412	21984	22025	22050	22099	23744	23748	23752	23792	23797	23858
	23862	23866	24661	25173	25725	26871	26880	28959	29209	29765	29846	30029
	31169	31309	31740	31748	31865	33245	33559	34704	35051	35055	35387	35543
	35651	35660	35779	35783	35802	35810	35897	35948	36010	36083	36088	36097
	36121	36248	36345	36350	36619	36796	36896	37182	37189	37208	37213	37219
	37336	37690	37709	37804	37827	37841	37864	37918	38035	38084	38098	38149
	38206	38219	38362	38449	38529	38534	38556	38763	38817	38894	38898	38906
	38924	38962	39034	39154	39944	39955	40858	41362	41385	41561	41643	41664
	41693	41698	41703	42041	42171	42508	42670	42683	47278	47570		
WMUX	3656 #	5381	5492	6135	6145	6695	7132	7955	8020	8036	8041	8046
	8982	8989	9112	9121	9401	9486	12633	14717	14745	14859	14875	15386
	16742	16755	16771	16782	16812	16827	16852	16868	16877	16883	16894	16917
	16933	16939	16950	16957	16980	17102	17254	17299	20156	20801	20950	21315
	21350	21412	21984	22025	22050	22099	23744	23748	23752	23792	23797	23858
	23862	23866	24661	25173	25725	26871	26880	28959	29209	29765	29846	30029
	31169	31309	31740	31748	31865	33245	33559	34704	35051	35055	35387	35543
	35651	35660	35779	35783	35802	35810	35897	35948	36010	36083	36088	36097
	36121	36248	36345	36350	36619	36796	36896	37182	37189	37208	37213	37219
	37336	37690	37709	37804	37827	37841	37864	37918	38035	38084	38098	38149
	38206	38219	38362	38449	38529	38534	38556	38763	38817	38894	38898	38906
	38924	38962	39034	39154	39944	39955	40858	41362	41385	41561	41643	41664
	41693	41698	41703	42041	42171	42508	42670	42683	47278	47570		

Location / Line Number Index

:Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
C 0000	43532:	43535:	43538:	43542:	43545:	43548:	43551:	43554:
C 0008	43557:	43560:	43563:	43566:	43569:	43572:	43575:	43578:
C 0010	43581:	43386:	43377:	43315:	43318:	43370:	43322:	43364:
C 0018	43380:	43373:	43325:	43367:	43389:	43392:	43309:	43312:
C 0020	43584:	43587:	43590:	43593:	43597:	43600:	43603:	43606:
C 0028	43609:	43612:	43615:	43618:	43621:	43624:	43627:	43630:
C 0030	43633:	43636:	43639:	43642:	43645:	43648:	43652:	43655:
C 0038	43658:	43661:	43664:	43667:	43670:	43673:	43676:	43679:
C 0040	43682:	43508:	43407:	43352:	43328:	43340:	43279:	43685:
C 0048	43688:	43514:	43413:	43358:	43334:	43346:	44009:	43691:
C 0050	43694:	43511:	43410:	43355:	43331:	43343:	43282:	43697:
C 0058	43700:	43517:	43416:	43361:	43337:	43349:	44012:	43703:
C 0060	43520:	44049:	43432:	43707:	43285:	43710:	43291:	44003:
C 0068	43713:	43716:	43719:	43722:	43725:	43728:	43731:	43734:
C 0070	43523:	44052:	43435:	43737:	43288:	43740:	43294:	44006:
C 0078	43743:	43746:	43749:	43752:	43755:	43758:	43762:	43765:
C 0080	43471:	43474:	43401:	43419:	43982:	43975:	43768:	43771:
C 0088	43438:	43441:	43404:	43422:	43985:	43978:	43774:	43777:
C 0090	43477:	43480:	43450:	43425:	43303:	43297:	43780:	43783:
C 0098	43444:	43447:	43453:	43428:	43306:	43300:	43786:	43789:
C 00A0	43483:	43487:	43526:	43273:	43792:	43499:	43795:	43798:
C 00A8	44030:	44033:	43529:	43276:	43801:	43496:	43804:	43807:
C 00B0	43490:	43493:	43997:	44043:	43505:	44027:	43810:	43813:
C 00B8	44037:	44040:	44000:	44046:	43502:	43817:	43820:	43823:
C 00C0	43826:	43829:	43988:	43383:	43456:	43832:	43994:	43835:
C 00C8	43838:	43841:	43844:	43847:	43850:	43853:	43856:	43859:
C 00D0	43862:	43865:	43868:	43872:	43875:	43878:	43881:	43884:
C 00D8	43887:	43890:	43893:	43896:	43899:	43902:	43905:	43908:
C 00E0	43911:	43459:	43914:	43462:	43917:	43465:	43920:	43468:
C 00E8	43923:	43927:	43930:	43933:	43936:	43939:	43942:	43945:
C 00F0	43991:	44015:	43395:	44016:	43948:	44021:	43398:	44024:
C 00F8	43951:	43954:	43957:	43960:	43963:	43966:	43969:	43972:

```
; This page intentionally left blank.
```

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
D 0000	22225:	22237:	39582:	22223:	20858:	20061:	37021:	37027:
D 0008	33381:	33383:	22227:	18002:	37023:	37025:	22233:	22247:
D 0010	20052:	20050:	20048:	20036:	20040:	20044:	20060:	20059:
D 0018	20038:	20046:	20041:	20045:	20054:	20055:	20034:	20035:
D 0020	33363:	33365:	33391:	33393:	33379:	33389:	33377:	33385:
D 0028	25497:	25489:	25505:	25509:	25499:	25491:	25501:	25503:
D 0030	20055:	20051:	10554:	10553:	33387:	33369:	33375:	33371:
D 0038	34782:	25495:	25493:	25507:	10585:	20023:	10578:	10595:
D 0040	17990:	17992:	18079:	18081:	18063:	18065:	18039:	18041:
D 0048	18016:	18022:	18020:	18030:	18006:	18033:	18026:	17984:
D 0050	18057:	18000:	18049:	18085:	18045:	18069:	18018:	39584:
D 0058	10517:	39566:	39588:	39590:	22229:	22231:	22243:	22245:
D 0060	17986:	17988:	18075:	18077:	18059:	18061:	18035:	18037:
D 0068	18008:	18014:	18012:	18028:	18004:	18032:	18024:	17982:
D 0070	18055:	17998:	18047:	18083:	18043:	18067:	18010:	39592:
D 0078	10525:	10526:	10565:	10564:	10544:	10581:	10577:	10594:
D 0080	10518:	10519:	10599:	10600:	10586:	10587:	10558:	10559:
D 0088	10533:	10534:	10527:	10528:	10608:	10609:	10572:	20056:
D 0090	10579:	10546:	10569:	10539:	10542:	10605:	10566:	10555:
D 0098	10549:	10550:	10583:	10584:	10597:	20021:	10575:	10592:
D 00A0	10522:	10523:	10603:	10604:	10590:	10591:	10562:	10563:
D 00A8	10537:	10538:	10531:	10532:	10612:	10613:	10574:	20058:
D 00B0	10582:	10548:	10571:	10541:	10545:	10607:	10568:	10557:
D 00B8	22221:	22219:	22239:	22241:	36411:	36409:	36413:	36415:
D 00C0	10520:	10521:	10601:	10602:	10588:	10589:	10560:	10561:
D 00C8	10535:	10536:	10529:	10530:	10610:	10611:	10573:	20057:
D 00D0	10580:	10547:	10570:	10540:	10543:	10606:	10567:	10556:
D 00D8	10524:	10598:	36419:	36417:	22235:	10596:	10576:	10593:
D 00E0	20030:	20026:	20032:	20029:	20031:	20027:	20033:	20028:
D 00E8	20043:	20042:	18837:	18835:	18827:	18829:	18831:	18833:
D 00F0	18839:	20022:	20025:	20024:	20062:	20063:	10551:	10552:
D 00F8	33367:	33373:	20854:	20856:	22249:	39576:	39578:	39580:
D 0100	44140:	44141:	44142:	44143:	44144:	44145:	44146:	44147:
D 0108	44148:	44149:	44150:	44151:	44152:	44153:	44154:	44155:
D 0110	44156:	44092:	44089:	44069:	44070:	44087:	44071:	44085:
D 0118	44090:	44088:	44072:	44086:	44093:	44094:	44067:	44068:
D 0120	44157:	44158:	44159:	44160:	44161:	44162:	44163:	44164:
D 0128	44165:	44166:	44167:	44168:	44169:	44170:	44171:	44172:
D 0130	44173:	44174:	44175:	44176:	44177:	44178:	44179:	44180:
D 0138	44181:	44182:	44183:	44184:	44185:	44186:	44187:	44188:
D 0140	44189:	44132:	44099:	44081:	44073:	44077:	44057:	44190:
D 0148	44191:	44134:	44101:	44083:	44075:	44079:	44296:	44192:
D 0150	44193:	44133:	44100:	44082:	44074:	44078:	44058:	44194:
D 0158	44195:	44135:	44102:	44084:	44076:	44080:	44297:	44196:
D 0160	44136:	44309:	44107:	44197:	44059:	44198:	44061:	44294:
D 0168	44199:	44200:	44201:	44202:	44203:	44204:	44205:	44206:
D 0170	44137:	44310:	44108:	44207:	44060:	44208:	44062:	44295:
D 0178	44209:	44210:	44211:	44212:	44213:	44214:	44215:	44216:
D 0180	44120:	44121:	44097:	44103:	44287:	44285:	44217:	44218:
D 0188	44109:	44110:	44098:	44104:	44288:	44286:	44219:	44220:
D 0190	44122:	44123:	44113:	44105:	44065:	44063:	44221:	44222:
D 0198	44111:	44112:	44114:	44106:	44066:	44064:	44223:	44224:
D 01A0	44124:	44125:	44138:	44055:	44225:	44129:	44226:	44227:

:Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
D 01A8	44303:	44304:	44139:	44056:	44228:	44128:	44229:	44230:
D 01B0	44126:	44127:	44292:	44307:	44131:	44302:	44231:	44232:
D 01B8	44305:	44306:	44293:	44308:	44130:	44233:	44234:	44235:
D 01C0	44236:	44237:	44289:	44091:	44115:	44238:	44291:	44239:
D 01C8	44240:	44241:	44242:	44243:	44244:	44245:	44246:	44247:
D 01D0	44248:	44249:	44250:	44251:	44252:	44253:	44254:	44255:
D 01D8	44256:	44257:	44258:	44259:	44260:	44261:	44262:	44263:
D 01E0	44264:	44116:	44265:	44117:	44266:	44118:	44267:	44119:
D 01E8	44268:	44269:	44270:	44271:	44272:	44273:	44274:	44275:
D 01F0	44290:	44298:	44095:	44299:	44276:	44300:	44096:	44301:
D 01F8	44277:	44278:	44279:	44280:	44281:	44282:	44283:	44284:

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
I 0000	22130:	22171:	39535:	22123:	20844:	20001:	36984:	37014:
I 0008	33314:	33321:	22137:	17705:	36994:	37004:	22157:	22206:
I 0010	19939:	19925:	19916:	19848:	19866:	19894:	19994:	19987:
I 0018	19857:	19907:	19873:	19900:	19952:	19959:	19835:	19842:
I 0020	33252:	33259:	33348:	33355:	33307:	33342:	33300:	33328:
I 0028	25441:	25413:	25468:	25482:	25448:	25420:	25455:	25461:
I 0030	19946:	19932:	10105:	10098:	33335:	33273:	33294:	33280:
I 0038	34773:	25434:	25427:	25475:	10318:	19759:	10270:	10387:
I 0040	17674:	17681:	17954:	17961:	17909:	17915:	17836:	17843:
I 0048	17753:	17774:	17767:	17802:	17719:	17815:	17788:	17653:
I 0050	17888:	17698:	17870:	17974:	17857:	17929:	17760:	39542:
I 0058	9850:	39549:	39556:	39562:	22144:	22151:	22192:	22199:
I 0060	17660:	17667:	17940:	17947:	17895:	17902:	17822:	17829:
I 0068	17726:	17747:	17740:	17795:	17712:	17808:	17781:	17646:
I 0070	17881:	17692:	17863:	17967:	17850:	17922:	17733:	39569:
I 0078	9905:	9912:	10180:	10173:	10036:	10290:	10263:	10380:
I 0080	9857:	9864:	10414:	10421:	10325:	10332:	10132:	10139:
I 0088	9960:	9967:	9919:	9926:	10476:	10483:	10228:	19966:
I 0090	10277:	10050:	10208:	10002:	10022:	10455:	10187:	10112:
I 0098	10070:	10077:	10304:	10311:	10400:	19745:	10249:	10366:
I 00A0	9885:	9892:	10442:	10448:	10352:	10359:	10160:	10167:
I 00A8	9988:	9995:	9947:	9953:	10503:	10510:	10242:	19980:
I 00B0	10297:	10063:	10222:	10015:	10043:	10469:	10201:	10125:
I 00B8	22116:	22109:	22178:	22185:	36374:	36367:	36381:	36388:
I 00C0	9871:	9878:	10428:	10435:	10338:	10345:	10146:	10153:
I 00C8	9974:	9981:	9933:	9940:	10490:	10497:	10235:	19973:
I 00D0	10283:	10057:	10215:	10008:	10029:	10462:	10194:	10118:
I 00D8	9898:	10407:	36402:	36395:	22164:	10393:	10256:	10373:
I 00E0	19807:	19780:	19821:	19800:	19814:	19787:	19828:	19793:
I 00E8	19887:	19880:	18813:	18806:	18778:	18785:	18792:	18799:
I 00F0	18820:	19752:	19773:	19766:	20007:	20014:	10084:	10091:
I 00F8	33266:	33287:	20826:	20835:	22212:	39514:	39521:	39528:

; This page intentionally left blank.

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
0 0000	22133:	22174:	39538:	22126:	20847:	20004:	36987:	37017:
0 0008	33317:	33324:	22140:	17708:	36997:	37007:	22160:	22209:
0 0010	19942:	19928:	19919:	19851:	19869:	19897:	19997:	19990:
0 0018	19860:	19910:	19876:	19903:	19955:	19962:	19838:	19845:
0 0020	33255:	33262:	33351:	33358:	33310:	33345:	33303:	33331:
0 0028	25444:	25416:	25471:	25485:	25451:	25423:	25458:	25464:
0 0030	19949:	19935:	10108:	10101:	33338:	33276:	33297:	33283:
0 0038	34776:	25437:	25430:	25478:	10321:	19762:	10273:	10390:
0 0040	17677:	17684:	17957:	17964:	17912:	17918:	17839:	17846:
0 0048	17756:	17777:	17770:	17805:	17722:	17818:	17791:	17656:
0 0050	17891:	17701:	17873:	17977:	17860:	17932:	17763:	39545:
0 0058	9853:	39552:	39559:	39565:	22147:	22154:	22195:	22202:
0 0060	17663:	17670:	17943:	17950:	17898:	17905:	17825:	17832:
0 0068	17729:	17750:	17743:	17798:	17715:	17811:	17784:	17649:
0 0070	17884:	17695:	17866:	17970:	17853:	17925:	17736:	39572:
0 0078	9908:	9915:	10183:	10176:	10039:	10293:	10266:	10383:
0 0080	9860:	9867:	10417:	10424:	10328:	10335:	10135:	10142:
0 0088	9963:	9970:	9922:	9929:	10479:	10486:	10231:	19969:
0 0090	10280:	10053:	10211:	10005:	10025:	10458:	10190:	10115:
0 0098	10073:	10080:	10307:	10314:	10403:	19748:	10252:	10369:
0 00A0	9888:	9895:	10445:	10451:	10355:	10362:	10163:	10170:
0 00A8	9991:	9998:	9950:	9956:	10506:	10513:	10245:	19983:
0 00B0	10300:	10066:	10225:	10018:	10046:	10472:	10204:	10128:
0 00B8	22119:	22112:	22181:	22188:	36377:	36370:	36384:	36391:
0 00C0	9874:	9881:	10431:	10438:	10341:	10348:	10149:	10156:
0 00C8	9977:	9984:	9936:	9943:	10493:	10500:	10238:	19976:
0 00D0	10286:	10060:	10218:	10011:	10032:	10465:	10197:	10121:
0 00D8	9901:	10410:	36405:	36398:	22167:	10396:	10259:	10376:
0 00E0	19810:	19783:	19824:	19803:	19817:	19790:	19831:	19796:
0 00E8	19890:	19883:	18816:	18809:	18781:	18788:	18795:	18802:
0 00F0	18823:	19755:	19776:	19769:	20010:	20017:	10087:	10094:
0 00F8	33269:	33290:	20829:	20838:	22215:	39517:	39524:	39531:

; This page intentionally left blank.

; Location / Line Number Index

;Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 0000	5324:	20972:	37887:	18521=	9019=	9024=	9027=	18524=
U 0008	37144:	9112=	8240	9116=	38913:	38918:	38616=	8492=
U 0010	42398:	38529:	38534:	40986=	38807:	38579:	6122:	41021=
U 0018	9095=	9100=	9104=	41059=	8977=	8982=	38622=	11226=
U 0020	37635:	39644:	39875:	38385:	39674:	39684:	39722:	39736:
U 0028	37629:	39998:	39909:	39965:	38717:	38414:	38351:	38346:
U 0030	9149=	9154=	9159=	9164=	39678:	39688:	18448=	18451=
U 0038	37099:	37105:	37110:	37115:	37121:	37125:	37131:	37137:
U 0040	8905=	8910=	18699=	18703=	8916=	8920=	9191=	9195=
U 0048	9170=	9175=	9180=	9185=	9199=	9203=	8245	8603=
U 0050	8606=	9336=	8261	9341=	9209=	9214=	9219=	9224=
U 0058	18868=	18873=	18878=	18883=	18888=	18893=	18898=	18903=
U 0060	10740=	41097=	10748=	10754=	9378=	9383=	9388=	41135=
U 0068	11552=	11558=	11564=	41171=	9230=	9234=	8266	8692=
U 0070	8695=	41221=	9238=	9242=	11641=	11647=	11651=	41247=
U 0078	18927=	18931=	18935=	18939=	18943=	18947=	18951=	18955=
U 0080	12829=	12834=	12840=	12846=	12852=	12858=	12866=	12870=
U 0088	11978=	11984=	11988=	41481=	9306=	9310=	8278	8777=
U 0090	8780=	41520=	9666=	9670=	13048=	13053=	13058=	13064=
U 0098	18971=	18976=	18981=	18986=	18991=	18996=	19001=	19005=
U 00A0	13450=	13456=	13462=	13469=	13473=	41587=	10978=	10983=
U 00A8	13153=	13160=	13169=	41679=	11253=	11262=	8292	8821=
U 00B0	8824=	41683=	11322=	11329=	13229=	13236=	13241=	13248=
U 00B8	13763=	13768=	13774=	41742=	11337=	11340=	19042=	19048=
U 00C0	13863=	13867=	13872=	13878=	13885=	41814=	11441=	11448=
U 00C8	13782=	13788=	13793=	13800=	11630=	11634=	8307	8866=
U 00D0	8869=	42460=	14937=	14942=	14306=	14311=	14317=	42531=
U 00D8	14325=	14331=	14336=	14343=	16735=	16742=	19053=	19061=
U 00E0	15301=	15305=	15310=	15315=	15322=	8333	16748=	16755=
U 00E8	14420=	14426=	14431=	14437=	16762=	16771=	8351	18199=
U 00F0	18202=	8357	16776=	16782=	16902=	16907=	16912=	16917=
U 00F8	16965=	16970=	16975=	16980=	16804=	16812=	19065=	19071=
U 0100	44349:	44354:	44358:	44364:	44369:	44374:	44379:	44384:
U 0108	44388:	44393:	44398:	44403:	44406:	44410:	44416:	38610=
U 0110	44821:	44826:	44829:	44832:	44837:	44842:	44847:	44852:
U 0118	44857:	44861:	44866:	44871:	44874:	44878:	44884:	38627=
U 0120	44929:	44934:	44938:	44941:	44946:	44951:	44956:	44960:
U 0128	44965:	44970:	44974:	44979:	44982:	44986:	44992:	38632=
U 0130	45058:	45063:	45067:	45070:	45075:	45080:	45085:	45089:
U 0138	45094:	45098:	45103:	45108:	45111:	45115:	45121:	42188=
U 0140	45169:	45174:	45177:	45180:	45187:	45194:	45201:	45207:
U 0148	45212:	45217:	45222:	45227:	45230:	45235:	42336=	42428=
U 0150	45276:	45281:	45284:	45287:	45292:	45297:	45302:	45306:
U 0158	45310:	45315:	45320:	45325:	45328:	45332:	42346=	42479=
U 0160	45356:	45361:	45365:	45370:	45375:	45380:	45385:	45390:
U 0168	45394:	45399:	45404:	45409:	45412:	45416:	45420:	42503=
U 0170	45443:	45448:	45452:	45457:	45462:	45467:	45472:	45477:
U 0178	45481:	45486:	45491:	45496:	45499:	45503:	45511:	42645=
U 0180	45590:	45595:	45599:	45604:	45608:	45613:	45618:	45623:
U 0188	45627:	45632:	45637:	45642:	45645:	45649:	45654:	46903=
U 0190	45703:	45708:	45711:	45714:	45718:	45723:	45728:	45733:
U 0198	45737:	45742:	45747:	45752:	45755:	45759:	45767:	47077=
U 01A0	45818:	45823:	45826:	45829:	45834:	45839:	45844:	45849:

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 01A8	45854:	45858:	45863:	45868:	45871:	45875:	45881:	8363
U 01B0	45900:	45905:	45910:	45916:	45922:	45928:	45934:	45939:
U 01B8	45944:	45950:	45956:	45961:	45964:	45969:	45974:	8368
U 01C0	46009:	46014:	46017:	46020:	46025:	46030:	46035:	46040:
U 01C8	46044:	46049:	46054:	46059:	46062:	46066:	46071:	8373
U 01D0	8378	14635=	14641=	14646=	14652=	14657=	14662=	14666=
U 01D8	14673=	14678=	16818=	16827=	16997=	17000=	8388	17005=
U 01E0	15680=	15685=	15690=	15695=	15699=	8405	16845=	16852=
U 01E8	17041=	17045=	17049=	17056=	16877=	16883=	8436	18218=
U 01F0	18221=	8441	16888=	16894=	17165=	17169=	17173=	17177=
U 01F8	19607=	19611=	19615=	19619=	19623=	19627=	19631=	19635=
U 0200	16200=	16205=	16923=	16927=	17246=	17249=	8455	17254=
U 0208	17286=	17290=	17295=	17299=	16933=	8467	16939=	18275=
U 0210	18279=	8506	16950=	16957=	17313=	17318=	17323=	17328=
U 0218	19659=	19664=	19669=	19674=	19679=	19684=	19689=	19693=
U 0220	16225=	8522	16987=	16991=	17451=	17457=	17463=	17466=
U 0228	18396=	18402=	8528	18406=	17072=	17076=	8553	18287:
U 0230	18291:	8558	39475:	39479:	18410=	18414=	18418=	8590
U 0238	17010=	17014=	17020=	17026=	17034=	8596	17118=	17123=
U 0240	17196=	17200=	17204=	8620	18461=	18466=	18471=	8636
U 0248	18476=	18479=	18483=	18487=	17156=	17160=	8640	18316=
U 0250	18321=	8679	17234=	17238=	18491=	18494=	18498=	18502=
U 0258	17211=	17215=	17218=	8685	18506=	18509=	18513=	18517=
U 0260	17261=	17266=	17270=	17275=	17279=	8709	18231=	18236=
U 0268	18739=	18743=	18747=	18751=	18241=	18245=	8727	18331=
U 0270	18335=	8732	18708=	18711=	22605=	22611=	22615=	8763
U 0278	17602=	8770	18762=	18765=	26096=	26101=	26104=	8807
U 0280	18652=	18656=	18769=	18772=	18660=	18665=	19144=	19150=
U 0288	26419=	26425=	26429=	8814	19337=	19343=	8851	19280=
U 0290	19284=	8858	21386=	21390=	26436=	26441=	26444=	8899
U 0298	18672=	18675=	18679=	18682=	18685=	18688=	21396=	21400=
U 02A0	19733:	16462:	21518=	21521=	30333=	30338=	30342=	8972
U 02A8	30435:	30440:	30443:	38910:	21585=	21589=	9121	19515:
U 02B0	19519:	9125	21651=	21655=	31423=	31427=	31432=	31436=
U 02B8	20531=	9271	21764=	21768=	33484=	33489=	33492=	9276
U 02C0	14949=	14956=	14961=	14966=	14973=	14982=	14987=	14993=
U 02C8	20901=	9281	21854=	21858=	40991=	9287	40996=	41001=
U 02D0	42736=	42740=	42745=	42749=	42754=	42759=	42764=	42769=
U 02D8	42774=	42779=	42785=	42789=	42794=	42799=	42804=	42809=
U 02E0	42883=	42888=	42893=	42897=	42902=	42907=	42913=	42918=
U 02E8	42922=	42928=	42935=	42939=	42944=	42949=	42955=	42959=
U 02F0	43024=	43029=	43034=	43038=	43043=	43048=	43054=	43059=
U 02F8	43063=	43069=	43076=	43080=	43084=	43088=	43094=	43098=
U 0300	20914=	9292	22425=	22429=	41027=	9297	41032=	41038=
U 0308	20930=	9301	22663=	22666=	41065=	10679	41070=	41076=
U 0310	20950=	10684	25608=	25612=	41103=	10733	41108=	41114=
U 0318	21007=	21012=	21017=	21022=	21026=	10804	29459=	29463=
U 0320	22705=	22712=	22717=	22722=	22725=	10808	31412=	31416=
U 0328	24735=	10910	36466=	36470=	41140=	10915	41145=	41150=
U 0330	25015=	11093	36550=	36554=	41176=	11122	41181=	41186=
U 0338	25196=	11353	41802=	41807=	41821=	11571	41826=	41831=
U 0340	25276=	11756	41860=	41863=	42192=	12035	42196=	42200=
U 0348	25357=	12042	41892=	41895=	41923=	41926=	42249=	42252=

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 0350	29074=	12046	42282=	42285=	42315=	42318=	13146	13222
U 0358	30025=	15471	16414	16435	16723	16728	16787	16792
U 0360	30091=	30096=	30101=	30105=	30109=	30114=	30118=	16797
U 0368	30182=	30187=	30190=	16834	16839	16860	16868	16943
U 0370	30193:	30197:	30202:	30206:	30211:	30215:	38354:	17085
U 0378	31214:	17091	17096	39123:	17221	17225	17306	17335
U 0380	32029=	17443	18250	18431	18443	18455	18692	18695
U 0388	34654=	18715	18718	18723	18726	18729	18734	18754
U 0390	34953=	34957=	34961=	34964=	34968=	18757	19022	19154
U 0398	35747=	35751=	35756=	35760=	35764=	19237	19242	19330
U 03A0	36667=	19382	19444	19553	19715	20099	21099	21201
U 03A8	36841=	21350	21434	21513	21758	21848	22419	22498
U 03B0	38706=	22502	22557	22561	22597	22654	22699	25602
U 03B8	39329=	26090	29810	29814	30085	30176	30326	30429
U 03C0	39205:	33467	39224:	39228:	39217:	35741	36228	36233
U 03C8	36238	36242	41007	41045	41083	41121	41156	41193
U 03D0	41206	41276	41282	41297	41400	41500	41539	41561
U 03D8	41619	41643	41664	41755	41837	41854	41886	41917
U 03E0	41948	42077	42204	42243	42276	42309	42369	42465
U 03E8	42559	42598	42292:	22731:	24433:	24437:	41316:	47393:
U 03F0	39264:	9445:	9449:	9454:	9405:	9410:	9416:	9420:
U 03F8	37641:	37647:	41991:	41996:	39944:	39949:	39955:	39960:
U 0400	44469=	44473=	44477=	44480=	44485=	44489=	44493=	44497=
U 0408	44501=	44505=	44508=	44512=	44515=	44518=	44523=	8321
U 0410	44564=	44568=	44572=	44575=	44580=	44585=	44589=	44593=
U 0418	44597=	44601=	44604=	44608=	44611=	44614=	44619=	8420
U 0420	44654=	44658=	44662=	44665=	44670=	44675=	44679=	44683=
U 0428	44687=	44691=	44694=	44698=	44701=	44704=	44709=	8460
U 0430	44743=	44747=	44751=	44754=	44759=	44763=	44767=	44771=
U 0438	44775=	44779=	44782=	44786=	44789=	44792=	9677=	9682=
U 0440	9039=	8472	9581=	9045=	9050=	8946=	9585=	8950=
U 0448	9554=	9559=	9565=	9570=	9574=	9031=	8511	9034=
U 0450	8927=	8932=	8937=	8942=	9315=	9319=	9324=	9329=
U 0458	9435:	9441:	9425:	9430:	9459:	9465:	9471:	8564:
U 0460	9481=	9486=	8653	9490=	8737	9502=	9072	9506=
U 0468	9515=	9519=	9523=	9527=	9477	9603=	9511	9608=
U 0470	9591=	9596=	9625=	9630=	9600=	9637=	9634=	9642=
U 0478	9687=	9690=	9694=	9698=	9703=	9707=	9711=	9760
U 0480	9737=	9743=	9750=	9756=	9765	9775=	9778=	9782=
U 0488	9770	9788=	9805	9791=	44426	9796=	44432	9800=
U 0490	9809=	9813=	9817=	9821=	44438	9826=	44444	9830=
U 0498	9835=	9840=	44449	9844=	8297=	8301=	8312=	8316=
U 04A0	8411=	8416=	8498=	8501=	8532=	8536=	8645=	8648=
U 04A8	9055=	9059=	9063=	9067=	9396=	9401=	9495=	9498=
U 04B0	9614=	9619=	9648=	9653=	9717=	9721=	9727=	9731=
U 04B8	45670=	45674=	44456	44529	44534	44539	44545	44625
U 04C0	44630	44635	44641	44714	44719	44724	44730	44805:
U 04C8	44891	44898	44904	44909	44915	44998	45006	45012
U 04D0	45017	45022	45027	45033	45040	45126	45131	45138
U 04D8	45143	45149	45240	45247	45252	45338	45425	45517
U 04E0	45525	45530	45534	45542	45548	45554	45559	45565
U 04E8	45570	45659	45664	45678	45683	45775	45781	45786
U 04F0	45790	45795	45802	45978	45983	45987	45991	46079

;Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 04F8	46085	11683=	10831	11691=	10689=	10694=	10699=	10704=
U 0500	10760=	10767=	10770=	10777=	11026=	11029=	11899=	11903=
U 0508	11910=	11917=	12513=	12519=	10813=	10817=	10822=	10825=
U 0510	12333=	12338=	12343=	11116	12348=	11804=	11170	11808=
U 0518	11181	12380=	11493=	12385=	12390=	12365=	11499=	12368=
U 0520	13573=	13578=	13582=	13588=	13592=	13596=	13603=	11345
U 0528	14209=	14214=	14220=	14223=	10864=	10868=	10875=	10879=
U 0530	11930=	14735=	11935=	14740=	14745=	14751=	15220=	15226=
U 0538	14837=	14843=	14846=	14854=	14859=	11387	14869=	14875=
U 0540	15188=	15192=	15198=	15202=	15207=	12396=	11761	12400=
U 0548	11813	15766=	15771=	15777=	15783=	15789=	15794=	11825
U 0550	11890	15889=	15894=	15900=	15905=	15911=	15837=	15842=
U 0558	11034=	11037=	16270=	16274=	16278=	16284=	16287=	16290=
U 0560	16483=	16488=	16493=	16497=	10922=	10926=	10930=	10934=
U 0568	16512=	16516=	16521=	16525=	16528=	16534=	16539=	11923
U 0570	10988=	10992=	10996=	11002=	11007=	11012=	11016=	11021=
U 0578	11098=	11103=	11107=	11110=	11126=	11132=	11136=	11141=
U 0580	11230=	11235=	11240=	11245=	11359=	11364=	11369=	11374=
U 0588	11765=	11771=	11777=	11782=	11790=	11795=	11799=	12052
U 0590	12058=	12063=	12066=	12070=	12132=	12137=	12142=	12146=
U 0598	12151=	12157=	12162=	12167=	12224=	12229=	12233=	12238=
U 05A0	12406=	12128	12372	12411=	12562=	12566=	12570=	12607
U 05A8	12587=	12591=	12594=	11658=	12612=	12616=	12621=	11663=
U 05B0	12700=	12706=	12712=	12716=	12721=	12726=	12731=	12734=
U 05B8	12954=	12959=	12964=	12969=	12974=	12979=	12985=	12990=
U 05C0	13081=	13085=	16125=	16129=	13309=	13314=	13320=	13325=
U 05C8	13352=	13357=	13362=	13367=	13611=	12628	12633	13615=
U 05D0	13625=	12637	13630=	13636=	12878=	13665=	12883=	13670=
U 05D8	13070=	13684=	13075=	13688=	13333=	13699=	13337=	13702=
U 05E0	13343=	13712=	13348=	13716=	13946=	13951=	13956=	13960=
U 05E8	13965=	13970=	13974=	13977=	13987=	14039=	13992=	14044=
U 05F0	14804=	14229=	14235=	14238=	14809=	14249=	14255=	14259=
U 05F8	14520=	14471=	14523=	14475=	14501=	14505=	14509=	14513=
U 0600	15819=	14684=	15824=	14690=	14697=	14702=	14705=	12694
U 0608	14713=	14717=	14723=	12745	14769=	14774=	14779=	14784=
U 0610	12887	15001=	15008=	15016=	15110=	15115=	15120=	15123=
U 0618	16137=	15340=	16141=	15343=	15348=	15353=	15358=	15362=
U 0620	17589=	15372=	17592=	15376=	15414=	15418=	15422=	15426=
U 0628	15478=	15482=	15489=	15496=	15527=	15532=	15538=	15544=
U 0630	15707=	15711=	15714=	14816=	15737=	15742=	15747=	14822=
U 0638	12892	15801:	12950	15807:	15988=	15992=	15997=	13371
U 0640	11146=	11149=	16214=	16219=	16235=	16239=	16242=	13375
U 0648	16248=	16251=	16256=	16259=	16422=	16425=	16430=	13480
U 0650	16442=	16449=	16453=	16456=	13618	16474=	13641	16478=
U 0658	13646	16570:	13674	16574:	11154=	11158=	16611=	16618=
U 0660	13678	16630=	13693	16633=	17500=	17504=	17508=	13707
U 0668	17531=	17536=	17541=	13982	17547=	17552=	14077	17557=
U 0670	14093	17626=	14098	17630=	11163=	11166=	11174=	11177=
U 0678	11380=	11383=	11455=	11459=	11467=	11473=	11482=	11489=
U 0680	11672=	11678=	11699=	11706=	11712=	11715=	11817=	11820=
U 0688	11882=	11885=	12354=	12360=	12527=	12534=	12542=	12545=
U 0690	12550=	12556=	12576=	12580=	12599=	12603=	12642=	12646=
U 0698	12738=	12741=	13091=	13095=	13653=	13657=	14049=	14053=

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 06A0	14059=	14063=	14070=	14073=	14083=	14088=	14113=	14118=
U 06A8	14451=	14455=	14462=	14466=	14480=	14484=	14490=	14495=
U 06B0	14758=	14763=	14791=	14795=	15079=	15084=	15095=	15098=
U 06B8	15137=	15141=	15153=	15157=	15177=	15183=	15212=	15216=
U 06C0	15163=	15239=	15330=	15335=	15166=	15243=	15395=	15401=
U 06C8	15505=	15511=	15516=	15520=	15726=	15731=	15755=	15760=
U 06D0	15876=	15880=	15919=	15923=	16001=	16006=	16027=	16030=
U 06D8	16060=	16064=	16502=	16506=	16557=	16563=	17488=	17494=
U 06E0	17610=	17613=	14103	14108	14242	14442	14446	14799
U 06E8	14830	15073	15088	15104	15127	15132	15147	15170
U 06F0	15232	15367	15380	15386:	15390:	15551	15719	15812
U 06F8	15830	16010	16015	16021	16035	16041	16047	16053
U 0700	16069	16074	16114	16120	16230	16263	16294	16468:
U 0708	16544	16551	16578:	16584	16624	16638	16643	16650
U 0710	17470	17478	17482	17512	17517	17522	17566	17578
U 0718	17582	17562:	17596	17616	17621	17634	17640	15407:
U 0720	17474:	5519:	18910=	18915=	19093=	19098=	19083=	19087=
U 0728	19102=	19105=	19109=	19113=	19117=	19121=	19125=	19129=
U 0730	19010	19311=	19077	19316=	19160=	19163=	19174=	19178=
U 0738	19193=	19196=	19200=	19204=	19208=	19212=	19216=	19220=
U 0740	19410=	19414=	19418=	19421=	19424=	19427=	19184=	19189=
U 0748	19461=	19465=	19470=	19474=	19479=	19484=	19250=	19254=
U 0750	19167	19322=	19304	19323=	19349	19354=	19400	19359=
U 0758	19534=	19540=	19546=	19405	19293=	19299=	19391=	19395=
U 0760	19489=	19493=	19572=	19575=	19590=	19595=	19451	19456
U 0768	19525	19529	19559	19566	19578	19583	19640	19645
U 0770	19698	19702	19719	19723	19728	19739	39233:	47515
U 0778	42211:	47520	42217:	42222:	42226:	47570	47615=	47621=
U 0780	47556=	47561=	48008=	48014=	47566=	47575	48018=	48022=
U 0788	47466=	47470=	47474=	47478=	47495=	47500=	47505=	47510=
U 0790	47541=	47546=	47551=	47655	47626=	47631=	47636=	47641=
U 0798	47676=	47681=	47686=	47691=	47733=	47737=	47740=	47661
U 07A0	47759=	47763=	47767=	47697	47773=	47777=	47782=	47798
U 07A8	47815	47954=	47960=	47963=	47970=	47975=	47979=	47984=
U 07B0	47828	48044=	48049=	48053=	47647=	47651=	47667=	47671=
U 07B8	47722=	47726=	47745=	47752=	47789=	47793=	47805=	47810=
U 07C0	47937:	47943:	47997=	48001=	48029=	48033=	47832	47835
U 07C8	47840	47844	47848	47852	47856	47859	47862	47868
U 07D0	47873	47877	47882	47886	47889	47893	47899	47904
U 07D8	47909	47947	47988	47992	48038	5523:	5605:	5943:
U 07E0	5948:	5951:	5954:	5957:	5960:	20147:	17081:	20150:
U 07E8	35567:	35571:	35576:	39210:	17102:	39269:	39288:	39295:
U 07F0	8986:	8989:	48084	48087	48090	48093	28490:	28494:
U 07F8	31166:	31169:	31224:	44551:	17107:	17111:	39239:	39245:
U 0800	5335=	5339=	5362=	5366=	5371=	5376=	5381=	5385=
U 0808	5389=	5393=	5399=	5403=	5427=	5432=	5435=	5328
U 0810	24680:	5438	24956:	5448	5455=	5459=	5463=	5468=
U 0818	5452	5343=	5485	5346=	5349=	5354=	5358=	5488
U 0820	29992:	5656:	5409=	5413=	5659:	5492	5648:	5651:
U 0828	5560=	5974=	5564=	5515	5569=	5978=	5442=	5445=
U 0830	5472=	5477=	5480=	5526	5547=	5553=	5556=	5540
U 0838	5574=	5601	5579=	5583=	5608	5587=	5591=	5596=
U 0840	5630=	5636=	5640=	5611	5614	5694=	5699=	5702=

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 0848	5913=	5920=	5924=	5927=	5617	5992=	5998=	6001=
U 0850	5504=	5508=	5531=	5535=	5672=	5676=	5733=	5737=
U 0858	5742=	5745=	5750=	5754=	5762=	5765=	5769=	5772=
U 0860	5777=	5780=	5791=	5796=	5807=	5810=	5814=	5818=
U 0868	5899=	5904=	5934=	5940=	5643	5662	5664	5667
U 0870	5682	5687	5712	5716	5723	5726	5785	5800
U 0878	5824	5876	5882	5888	5892	5929	5963	5968
U 0880	29660:	29664:	6007	6128	6132:	6135	7042=	7045=
U 0888	6167=	6171=	6174=	7975=	6156=	6160=	6163=	7978=
U 0890	6285=	6289=	6293=	6139	6244=	6248=	6251=	6256=
U 0898	6261=	6265=	6270=	6274=	6142	7110=	6277=	7113=
U 08A0	6508=	6513=	6517=	6520=	6523=	6145	6528=	6531=
U 08A8	6299=	6303=	6307=	6178	6149=	6152=	6281	6221:
U 08B0	6648=	6653=	6657=	6662=	6667=	6671=	6604	6675=
U 08B8	6679=	6682=	6686=	6690=	6695=	6698=	6702=	6608
U 08C0	6225=	6229=	6779=	6783=	6787=	6791=	8041=	8046=
U 08C8	6796=	7184=	6621	7187=	6313=	6317=	6321=	6749
U 08D0	6236=	6240=	6900=	6904=	6907=	6911=	6425=	6428=
U 08D8	6914=	7227=	6769	7231=	6327=	6331=	6335=	6773
U 08E0	6536=	6542=	6545=	6548=	6552=	6800	6557=	6560=
U 08E8	6581=	6587=	6592=	6596=	6600=	7627=	6803	7630=
U 08F0	6806	7199=	7204=	7208=	7211=	7635=	6814	7639=
U 08F8	7264=	7268=	7272=	7276=	7280=	7284=	7287=	6818
U 0900	6341=	6345=	6349=	6822	6825	7676=	6811:	7679=
U 0908	7548=	7553=	7557=	7561=	7565=	6828	7569=	7573=
U 0910	7607=	7611=	7615=	7618=	7621=	7683=	6832	7686=
U 0918	8123=	7726=	6836	7729=	8128=	8132=	8136=	8139=
U 0920	6355=	6359=	6363=	8058=	6369=	6373=	6377=	8063=
U 0928	6383=	6387=	6391=	6886	6397=	6401=	6405=	6889
U 0930	6411=	6415=	6419=	6892	6456=	6459=	6464=	6896
U 0938	6753=	6757=	6761=	6765=	6872=	6876=	6880=	6883=
U 0940	6956=	6960=	6965=	6918	6922	6983=	6987=	6991=
U 0948	7306=	7310=	7314=	6926	7746=	7750=	7753=	6952
U 0950	7762=	7765=	7769=	6969	7021	7801=	7024	7805=
U 0958	7027	7820=	7048	7824=	7868=	7873=	7877=	7051
U 0960	7055	7912=	7066	7915=	8020=	8026=	8032=	8036=
U 0968	8089=	8094=	8097=	7078	8187:	8191:	8195:	8199:
U 0970	7090	8164=	7102	8168=	6433=	6437=	6467=	6471=
U 0978	6474=	6478=	6483=	6487=	6491=	6494=	6499=	6503=
U 0980	6613=	6617=	6743=	6746=	6840=	6843=	6858=	6861=
U 0988	6864=	6868=	6974=	6978=	7014=	7017=	7033=	7037=
U 0990	7059=	7062=	7071=	7074=	7083=	7086=	7095=	7098=
U 0998	7126=	7129=	7135=	7138=	7162=	7166=	7170=	7173=
U 09A0	7177=	7180=	7235=	7239=	7291=	7295=	7318=	7322=
U 09A8	7326=	7330=	7334=	7338=	7342=	7345=	7349=	7353=
U 09B0	7357=	7361=	7365=	7369=	7373=	7378=	7383=	7386=
U 09B8	7393=	7396=	7410=	7413=	7420=	7423=	7438=	7441=
U 09C0	7445=	7447=	7459=	7462=	7466=	7469=	7482=	7485=
U 09C8	7489=	7492=	7504=	7507=	7511=	7514=	7526=	7529=
U 09D0	7533=	7536=	7578=	7581=	7585=	7588=	7644=	7648=
U 09D8	7652=	7655=	7664=	7667=	7695=	7699=	7702=	7706=
U 09E0	7710=	7714=	7718=	7722=	7734=	7737=	7782=	7786=
U 09E8	7829=	7833=	7846=	7850=	7854=	7857=	7861=	7864=

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 09F0	7900=	7903=	7920=	7925=	7944:	7949:	7955=	7958=
U 09F8	7982=	7986=	8009=	8014=	8072=	8076=	8202=	8206=
U 0A00	7107	7116	7119	7122	7132	7190	7193	7215
U 0A08	7220	7223	7389	7398	7401	7405	7416	7426
U 0A10	7429	7433	7450	7454	7472	7476	7495	7499
U 0A18	7517	7521	7539	7543	7591	7595	7659	7671
U 0A20	7690	7741	7757	7789	7792	7796	7809	7812
U 0A28	7815	7895	7908	7963	7968	8052	8068	8100
U 0A30	8104	8143	8147	8150	8154	8174:	8178:	36914:
U 0A38	22463=	22468=	22965=	22969=	22474=	22478=	22482=	22486=
U 0A40	22509=	22516=	22527=	22533=	22537=	22454	22684=	22688=
U 0A48	23018=	23022=	23026=	23046=	23031=	23035=	23039=	23050=
U 0A50	23086=	23091=	22981=	22985=	22458	23096=	23354=	23358=
U 0A58	23155=	23159=	23163=	23103=	23168=	23172=	23176=	23107=
U 0A60	23894=	22541=	23898=	22547=	22436=	22440=	23465=	23469=
U 0A68	22569=	22576=	22583=	22586=	22447=	22450=	23570=	23574=
U 0A70	23726=	23731=	23735=	23739=	22633	23744=	23748=	23752=
U 0A78	23183=	23783=	23757=	23787=	23187=	23792=	23761=	23797=
U 0A80	23804=	23845=	23849=	23853=	23808=	23858=	23862=	23866=
U 0A88	23922=	23925=	23928=	23933=	23937=	25258=	22745	25263=
U 0A90	24463=	24468=	23198=	23201=	24472=	23129=	22748	23133=
U 0A98	22752	24526=	22761	24533=	24537=	24541=	23326=	23329=
U 0AA0	22765	24963=	22773	24970=	24974=	22781	24980=	24984=
U 0AA8	25178=	25308=	22785	25314=	25183=	22930=	22936=	22940=
U 0AB0	22789	23231=	22801	23234=	22805	23279=	22921	23283=
U 0AB8	23307=	23311=	23315=	22952	23450=	23454=	23458=	22955
U 0AC0	22624	22627=	23475=	23479=	22973	23484=	22976	23487=
U 0AC8	23681=	23685=	23688=	22988	22991	23835=	22994	23839=
U 0AD0	23008	23904=	23053	23909=	23056	23970=	23974=	23980=
U 0AD8	22638=	22643=	24002=	24005=	23111	24036=	23137	24041=
U 0AE0	23141	24074=	24078=	24083=	22674=	22677=	24089=	24093=
U 0AE8	23145	24125=	23191	24130=	23205	24202=	24206=	24210=
U 0AF0	22913=	22917=	24216=	24221=	23237	24309=	24314=	24318=
U 0AF8	22946=	22949=	24324=	24328=	24367=	24372=	23245	24377=
U 0B00	24495=	24501=	24509=	23246	23261	24661=	23288	24665=
U 0B08	23292	24698=	23296	24701=	24770=	24775=	24778=	23320
U 0B10	23333	24816=	23437	24821=	24839=	24844=	24847=	23500
U 0B18	24855=	24860=	24863=	23509	23515	24930=	23521	24933=
U 0B20	25031=	25037=	25041=	23536	23563	25057=	23694	25061=
U 0B28	25071=	25076=	25080=	23768	23812	25110=	23943	25114=
U 0B30	25284=	25289=	25292=	23947	25364=	25368=	25372=	23957
U 0B38	23001=	23004=	23061=	23065=	23075=	23079=	23116=	23120=
U 0B40	23211=	23215=	23222=	23225=	23254=	23258=	23266=	23270=
U 0B48	23339=	23343=	23494=	23497=	23527=	23532=	23541=	23545=
U 0B50	23551=	23558=	23670=	23674=	23699=	23704=	23818=	23821=
U 0B58	23826=	23829=	23877=	23880=	23885=	23889=	24015=	24019=
U 0B60	24024=	24027=	24048=	24052=	24103=	24107=	24112=	24115=
U 0B68	24136=	24140=	24191=	24196=	24228=	24233=	24241=	24246=
U 0B70	24259=	24265=	24334=	24337=	24343=	24347=	24520:	24569:
U 0B78	24443=	24447=	24486=	24489=	24547=	24552=	24558=	24561=
U 0B80	24641=	24644=	24689=	24693=	24712=	24715=	24754=	24757=
U 0B88	24802=	24805=	24945=	24949=	24989=	24993=	25047=	25051=
U 0B90	25092=	25095=	25162=	25165=	25204=	25207=	25250=	25254=

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
0B98	24010	24062	24098	24251	24450	24457	24477	24481
0BA0	5422:	24423:	24565	24427:	24572	24576	24629	24632
0BA8	24635	24647	24651	24655	24669	24672	24675	24678
0BB0	24789:	24684	24704	24708	24718	24722	24741	24744
0BB8	24747	24760	24764	24782	24785	5418:	24793	24797
0BC0	24810	24825	24828	24831	24835	24869	24873	24877
0BC8	24880	24884	24925	24938	24952	24515:	24959	24998
0BD0	25002	25022	25025	25065	25086	25099	25104	25119
0BD8	25123	25128	25168	25173	25210	25215	25243	25296
0BE0	25299	25303	25319	25322	25326	25376	25379	25383
0BE8	25388	25395	25402	25406	6932:	6936:	5907:	44799:
0BF0	30296:	14728:	33953:	30781:	30772:	30775:	33956:	37841:
0BF8	31058:	31062:	37835:	37839:	37845:	22099:	47419:	22103:
0C00	20230	20235-	20240-	20244-	20249=	20253=	20258=	20262=
0C08	20266-	20270-	20275-	20279-	20284=	20288=	20293=	20297=
0C10	20301	20305-	20310-	20314-	20319=	20323=	20328=	20332=
0C18	20336-	20340-	20345=	20105	20124=	20129=	20112=	20118=
0C20	20155-	20140-	20350-	20156	20355=	20780=	20360=	20783=
0C28	20365-	20187	20370-	20197	20375=	20200	20379=	20203
0C30	20384-	20208	20389-	20211	20394=	20216	20399=	20462
0C38	20404-	20465	20409-	20469	20180=	20183=	20192=	20195=
0C40	20425-	20429-	20434-	20438=	20441=	20445=	20472=	20475=
0C48	20415-	20419-	20221=	20225=	20161=	20165=	20171=	20177=
0C50	20450-	20455-	20458-	20481	20487=	20491=	20495=	20499=
0C58	20731	20735-	20739=	20742=	20557=	20561=	20566=	20570=
0C60	20545-	20548-	20801=	20806=	20576=	20581=	20604=	20608=
0C68	20614	20618-	20625-	20627=	20633=	20637=	20643=	20647=
0C70	20653	20657	20663-	20667=	20672=	20676-	20682=	20686=
0C78	20692	20696	20702	20706=	20712=	20716=	20721=	20725=
0C80	20757	20761	20816	20820-	20504	20509	20514	20519
0C88	20536	20540	20555	20573	20586	20591	20594	20598
0C90	20746	20751	20765	20769	20774	20797	20811	20955
0C98	20966	21051-	20976	21055=	21123=	21128=	21133=	21137=
0CA0	21042	21046-	21102	21037-	21041=	21060=	21166	21064-
0CA8	21108	21112	21117	21206	21222=	21225=	21229=	21234=
0CB0	21216	21154-	21218	21160=	21259	21269-	21263	21272=
0CB8	20959=	20962=	21315=	21320=	21240=	21245=	21250=	21255=
0CC0	21439-	21445=	21446-	21276	21468=	21472=	21477=	21311
0CC8	21669	21675-	21678-	21324	21144=	21147=	21979=	21984-
0CD0	21210-	21215=	21283=	21286=	21408=	21412=	21450=	21454=
0CD8	21532-	21536=	21543=	21546=	21594=	21597=	21608=	21612=
0CE0	21702=	21705=	21720=	21723=	21776=	21780=	21791=	21794=
0CE8	21798	21801-	21820=	21824=	21871=	21874=	21885=	21889=
0CF0	21893=	21896-	21912=	21916=	21955=	21959=	21962=	21965=
0CF8	21989-	21994-	21999-	22004-	21527	21331	21355	21358
0D00	21362	21458	21461	21464	21483	21527	21539	21550
0D08	21554	21559	21600	21604	21615	21618	21621	21625
0D10	21659	21662	21665	21698	21708	21712	21716	21726
0D18	21730	21734	21777	21782	21787	21805	21808	21814
0D20	21861	21864	21868	21878	21881	21900	21903	21907
0D28	21920	21952	21969	21973	22046	22050	22054	22058
0D30	22093	29470=	29479	29474-	29546=	29550=	29556=	29560=
0D38	29651=	29567=	29656=	29571=	29703=	29694=	29707=	29698=

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 0D40	29982=	29736=	29986=	29739=	29754=	29758=	29762=	29765=
U 0D48	29873=	29877=	29882=	29886=	29488	29892=	29491	29895=
U 0D50	30002=	29931=	29495	29934=	30005=	29950=	29515	29954=
U 0D58	29530	29969=	29974=	29977=	29482=	29485=	29500=	29504=
U 0D60	29508=	29511=	29520=	29525=	29535=	29538=	29575=	29579=
U 0D68	29611=	29615=	29620=	29624=	29628=	29632=	29670=	29674=
U 0D70	29585=	29542	29590=	29636	29595=	29640	29600=	29645
U 0D78	29605=	29689	29680=	29684=	29712=	29717=	29723=	29727=
U 0D80	29825=	29828=	29843=	29846=	29856=	29860=	29907=	29912=
U 0D88	29921=	29924=	29731	29742	29745	29749	29768	29773
U 0D90	29776	29780	29818	29822	29831	29834	29836	29850
U 0D98	29869	29899	29903	29927	29938	34984	35169	35402=
U 0DA0	35919=	35923=	35927=	35931=	35935=	35939=	35944=	35180
U 0DA8	35214	34856=	35240	34859=	34838=	34842=	34846=	34850=
U 0DB0	35252	36027=	36031=	35265	36035=	36039=	34974=	34978=
U 0DB8	34864=	34867=	35268	34887=	34896=	34900=	35272	34891=
U 0DC0	34873=	34877=	34880=	34883=	35160=	35164=	35288	35257=
U 0DC8	34992=	34996=	35001=	35005=	35010=	35014=	35309	35261=
U 0DD0	35019=	35023=	35027=	35031=	35035=	35039=	35043=	35047=
U 0DD8	35051=	35055=	35059=	35064=	35067=	35071=	35075=	35079=
U 0DE0	35083=	35087=	35091=	35095=	35099	35103=	35107=	35111=
U 0DE8	35115=	34904=	35325	34908=	35328	34913=	35331	34916=
U 0DF0	35186=	35191=	35196=	35200=	35173=	35176=	35334	35119=
U 0DF8	35123=	35127=	35132=	35136=	35140=	35144=	35148=	35152=
U 0E00	35276=	35280=	35284=	35338	35353	35294=	35300=	35304=
U 0E08	36272=	36276=	36280=	36284=	36289=	35356	35343=	35349=
U 0E10	35771=	35775=	35779=	35783=	35787=	35791=	35795=	35798=
U 0E18	35802=	35806=	35810=	35814=	35818=	35822=	35826=	35830=
U 0E20	35834=	35838=	35842=	35846=	35850=	35853=	35857=	35861=
U 0E28	35865=	35360	35369=	35372=	35474=	35478=	35482=	35363
U 0E30	35205=	35209=	35485=	35489=	35218=	35222=	35376	35869=
U 0E38	35873=	35877=	35881=	35885=	35889=	35893=	35897=	35901=
U 0E40	35512=	35516=	35522=	35526=	35387	35532=	35536=	35540=
U 0E48	35585=	35590=	35593=	35390	35599=	35645=	35603=	35651=
U 0E50	35394	35666=	35399	35670=	35698=	35703=	35706=	35405
U 0E58	35408	36167=	35412	36174=	35427	36180=	35430	36184=
U 0E60	35433	36259=	35437	36263=	36300=	36304=	36308=	35441
U 0E68	35231=	35235=	35244=	35248=	35315=	35321=	35382=	35385=
U 0E70	35418=	35423=	35456=	35460=	35465=	35469=	35498=	35503=
U 0E78	35558=	35562=	35972=	35978=	35992=	35996=	36053=	36057=
U 0E80	36073=	36077=	36101:	36106:	36316=	36320=	36332=	36337=
U 0E88	36354=	36357=	35446	35451	35493	35507	35543	35547
U 0E90	35552	35580	35608	35612	35617	35621	35625	35905
U 0E98	35909	35914	35948	35951	35954	35958	35961	35964
U 0EA0	35968	35983	35987	36000	36004	36007	36010	36012
U 0EA8	36015	36021	36043	36047	36060	36063	36065	36069
U 0EB0	36083	36088	36092	36097	36121	36125	36128	36132
U 0EB8	36136	36139	36143	36147	36248	36251	36255	36268
U 0EC0	36293	36296	36312	36324	36328	36342	36345	36348
U 0EC8	36350	36361	8080:	8083:	37182:	37189:	37488=	37419=
U 0ED0	37864=	37868=	37873=	37878=	37882=	37326=	37205	37331=
U 0ED8	37357=	37361=	37208	37267=	37365=	37369=	37492=	37271=
U 0EE0	37962=	37965=	37969=	37973=	37977=	37981=	37423=	37428=

;Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 0EE8	37196=	37200=	37213	37275=	37228=	37231=	37219	37279=
U 0EF0	37237=	37242=	37248=	37252=	37375:	37380:	37256	38379:
U 0EF8	37672:	37675:	37679:	37283	37682:	37686:	37289	38358:
U 0F00	37384=	37397=	37388=	37400=	38971=	37341:	38975=	37346:
U 0F08	37437=	37441=	37312	38264=	37560=	37563=	37317	38267=
U 0F10	37994=	37998=	38002=	38006=	38011=	37410=	37322	37414=
U 0F18	39404=	39408=	39412=	39416=	39420=	38106=	37336	38109=
U 0F20	38419:	37349	37353	39504=	37850=	37853=	37391	39508=
U 0F28	37295=	37299=	37303=	37307=	37986=	37990=	38390:	38394:
U 0F30	38035=	38040=	38424:	38428:	38092=	38095=	37403	38457=
U 0F38	37479:	38298=	37406	38303=	38157=	38160=	37432	38461=
U 0F40	37509:	37483	38852=	38856=	38214=	38217=	37513	37537:
U 0F48	38483=	38487=	37586:	37690:	38655=	38659=	38434:	38438:
U 0F50	39154:	39158:	39163:	39167:	39174:	39176:	37658:	39181:
U 0F58	39185:	37694	38953=	38956=	37857	38405:	37908	38409:
U 0F60	37911	38473:	37914	38477:	38763:	38767:	38772:	38776:
U 0F68	39382=	39386=	39390=	39394=	37918	38884:	38065:	38888:
U 0F70	38894:	38898:	38902:	38906:	38962=	38965=	37958	38146:
U 0F78	38084	39398=	39351=	39355=	38088	37663=	38098	37666=
U 0F80	38924=	38928=	38933=	38102	38563:	38566:	39427=	39430=
U 0F88	39360=	39365=	38113	39369=	39463:	39469:	38811:	38817:
U 0F90	38149	38152	38164	38169	38174	38206	38210	38219
U 0F98	38222	38225	38229	38259	38202:	38272	38275	38279
U 0FA0	38283	38286	38289	38293	38367	38453	38556	38572
U 0FA8	38664	38712	38722	38757	38845	39191	38362:	39338
U 0FB0	38400:	38444:	38449:	38827:	39341	38467:	38493:	38834:
U 0FB8	38541:	38547:	38552:	38837:	39344	39347	38638:	38840:
U 0FC0	38999:	39003:	39008:	39011:	39016:	39020:	39025:	39029:
U 0FC8	39034:	39039:	39044:	39049:	39056:	39063:	39067:	39071:
U 0FD0	39075:	39079:	39083:	39087:	39091:	39099:	39106:	39113:
U 0FD8	39117:	38644:	38649:	39372	38699:	39376	39434	39437
U 0FE0	38823:	6709:	39147:	39493	39335:	39497	39501	6233:
U 0FE8	39442:	39446:	39450:	39454:	39457:	8992:	39120:	8996:
U 0FF0	29996:	36161:	28485:	31221:	39276:	39280:	36156:	48096
U 0FF8	38730:	38734:	48099	38740:	38746:	38752:	36112:	36116:
U 1000	25725=	25677=	25728=	25681=	25640=	25644=	25634	28167=
U 1008	25710=	25714=	27375=	27379=	25718=	25650	25792	28172=
U 1010	28178=	26125	27980=	28183=	28188=	25869=	27984=	25873=
U 1018	25617=	25620=	26150	25656=	25661=	25665=	26896=	26900=
U 1020	25994=	25997=	26003=	26006=	26009=	26014=	26017=	26021=
U 1028	26154	26480=	29339:	26483=	25969=	26486=	25973=	26489=
U 1030	29356:	26168	29361:	29366:	29370:	31902=	26172	31905=
U 1038	27161=	27165=	27169=	27173=	26189	27179=	27183=	27187=
U 1040	25797=	25802=	25807=	25811=	25626=	25630=	26193	29392=
U 1048	28824=	25859=	28828=	25863=	28832=	26198	28836=	29398=
U 1050	25687=	25692=	26212	29403=	27023=	32082=	27026=	32085=
U 1058	26871=	26876=	26880=	26217	26884=	32372=	26888=	32375=
U 1060	26224	25886=	28004=	25890=	25818=	25822=	28008=	28285=
U 1068	28298=	28303=	28307=	26237	28312=	25954=	26241	25957=
U 1070	30691=	30696=	30046=	28061=	26252	30700=	30050=	28065=
U 1078	26177=	26181=	26184=	26256	25828=	25833=	26274	28070=
U 1080	32396=	32400=	32404=	32408=	32412=	32416=	26277	32420=
U 1088	32423=	32426=	32537=	32541=	25838=	25843=	26280	32545=

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
1090	27705=	27711=	27720=	27724=	27728=	27732=	30596=	30601=
1098	31305=	27754=	27765=	27770=	31309=	27775=	27779=	26284
10A0	30127=	26361=	30132=	26364=	25848=	25851=	32055=	32059=
10A8	30138=	26370=	30142=	26373=	25896=	25901=	26293	32140=
10B0	26345	26589=	26593=	26598=	25907=	25913=	32064=	32068=
10B8	30532=	26616=	30536=	26620=	25921=	25927=	32072=	32143=
10C0	28039=	28044=	25932=	25936=	26349	28049=	26029=	26032=
10C8	29130:	29134:	29138:	29142:	29146:	26653=	26465	26656=
10D0	29214:	26475	29219:	29224:	29228:	26772=	26496	26775=
10D8	28208=	28213=	26111=	26115=	26509	28218=	26120=	26123=
10E0	30261=	30267=	30273=	30277=	26512	26824=	30281=	26827=
10E8	26131=	26136=	31588=	30352=	30356=	30360=	31591=	30365=
10F0	30849=	30853=	30857=	30861=	30865=	30869=	30875=	30879=
10F8	30938:	27014=	30942:	27017=	30948:	30952:	26143=	26146=
1100	31249=	31253=	26303=	26306=	31257=	27063=	31261=	27067=
1108	31740=	31745=	31748=	26516	31753=	27272=	31757=	27275=
1110	26318=	26322=	26578	32564=	32567=	32570=	32574=	26583
1118	32738=	32742=	26330=	26333=	26623	32747=	26338=	26341=
1120	32829=	32833=	32837=	32840=	32844=	32848=	26354=	26357=
1128	26637=	26641=	26645=	26627	33073=	33078=	33083=	33087=
1130	26671=	26675=	26679=	26683=	26700=	26705=	26710=	26715=
1138	26731=	26736=	26741=	26746=	27044=	27048=	27051=	26757
1140	27098=	27102=	27112=	26760	27118=	27122=	27126=	27130=
1148	27137=	27141=	27145=	26764	27254=	27259=	26767	27264=
1150	26808	27289=	26811	27293=	27333=	27337=	27344=	26815
1158	27499=	27503=	27506=	31650=	27513=	27517=	27520=	31653=
1160	27526=	27530=	27534=	27537=	26819	27835=	26835	27839=
1168	31675=	27845=	32435=	27849=	31678=	27855=	32438=	27859=
1170	27890=	26838	26842	27911=	27938=	27942=	27946=	27950=
1178	27960=	27965=	27971=	32456=	28014=	28018=	28022=	32459=
1180	28091=	28106=	28110=	27007	27029	28144=	27032	28147=
1188	27035	28253=	27039	28258=	28334=	28338=	28341=	27055
1190	28402=	28407=	28411=	27071	28453=	28458=	28462=	28466=
1198	27075	28568=	27197	28574=	28594=	28598=	28601=	27207
11A0	28633=	28637=	28641=	32475=	28714=	28718=	28721=	32478=
11A8	28730=	28734=	28738=	28742=	27217	28793=	27226	28797=
11B0	27229	29121=	27234	29124=	29246=	29250=	29254=	29258=
11B8	32491=	30230=	33147=	30233=	32494=	30286=	33150=	30290=
11C0	27298	30374=	27304	30380=	30384=	30388=	30392=	30396=
11C8	30457=	30461=	27325	30464=	30481=	30485=	30488=	27328
11D0	30581=	30586=	30590=	27417	30641=	30645=	30648=	27420
11D8	30565=	30670=	30674=	27423	30833=	30837=	30840=	27436
11E0	27450	31009=	27494	31014=	27540	31034=	31040=	31044=
11E8	31286=	31291=	31295=	31301=	27665	31505=	27668	31509=
11F0	27671	31535=	27674	31539=	31549=	31554=	31557=	27677
11F8	27692	31575=	27696	31579=	27701	31725=	27737	31728=
1200	32591=	32595=	32599=	32603=	32687=	27747	27820	32692=
1208	26452=	26456=	26469=	26472=	26501=	26504=	26604=	26608=
1210	26790=	26794=	26848=	26853=	27087=	27091=	27316=	27320=
1218	27357=	27361=	27428=	27431=	27442=	27446=	27455=	27459=
1220	27466=	27470=	27683=	27687=	27788=	27792=	27797=	27801=
1228	27814=	27817=	27916=	27919=	28117=	28121=	28245=	28249=
1230	28264=	28268=	28324=	28328=	28428=	28432=	28506=	28511=

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 1238	28534=	28538=	28559=	28563=	28584=	28589=	28619=	28623=
U 1240	28661=	28665=	28687=	28691=	28750=	28755=	28770=	28774=
U 1248	28779=	28783=	28806=	28810=	28942=	28946=	28979=	28983=
U 1250	29097=	29100=	29156=	29159=	29181=	29185=	29235=	29238=
U 1258	29269=	29272=	29281=	29285=	30237=	30240=	30244=	30249=
U 1260	30253=	30257=	30502=	30506=	30510=	30513=	30571=	30575=
U 1268	30611=	30614=	30620=	30626=	30632=	30636=	30655=	30659=
U 1270	30681=	30685=	30704=	30710=	30714=	30717=	30725=	30728=
U 1278	30738=	30742=	30747=	30750=	30756=	30760=	30787=	30790=
U 1280	30823=	30826=	30884=	30888=	30893=	30898=	30904=	30909=
U 1288	30915=	30919=	30959=	30963=	30974=	30978=	30984=	30987=
U 1290	30998=	31002=	31021=	31026=	31049=	31052=	31074:	31068:
U 1298	31116=	31119=	31134=	31139=	31442=	31448=	31461=	31464=
U 12A0	31470=	31474=	31495=	31498=	31523=	31527=	31604=	31608=
U 12A8	31618=	31622=	31635=	31639=	31659=	31662=	31704=	31708=
U 12B0	31715=	31719=	32090=	32093=	32197=	32200=	32211=	32214=
U 12B8	32234=	32237=	32248=	32251=	32271=	32274=	32285=	32288=
U 12C0	32551=	32556=	32625=	32629=	32635=	32639=	32645=	32649=
U 12C8	32655=	32659=	32664=	32667=	32720=	32723=	32791=	32794=
U 12D0	32815=	32819=	32858=	32862=	32867=	32871=	32880=	32884=
U 12D8	32889=	32894=	32903=	32907=	32916=	32920=	32926=	32930=
U 12E0	32938=	32942=	32948=	32952=	32965=	32970=	32979=	32984=
U 12E8	32992=	32996=	33002=	33006=	33027=	33031=	33037=	33041=
U 12F0	33047=	33051=	33061=	33065=	33093=	33096=	27823	27827
U 12F8	27868	27872	27876	27885	27923	27989	27992	27996
U 1300	28027	28030	28075	28079	28086	28125	28128	28131
U 1308	28134	28151	28155	28160	28241	28279	28347	28394
U 1310	28397	28418	28477:	28500:	28470	28522	28524	28528
U 1318	28554	28607	28670	28703	28725	28746	28765	28788
U 1320	28846	28849	28852	28855	28859	28866	28953	28956
U 1328	28959	28962	28971	28974	29080	29083	29086	29089
U 1330	29093	29103	29106	29109	29112	29116	29163	29167
U 1338	29176	29202:	29346:	29276	29334	29209:	29350	29376
U 1340	29381	29409	29412	29416	29421	29424	29428	30009
U 1348	30012	30019	30029	30033	30036	30039	30042	30054
U 1350	30222	30226	30346	30370	30401	30447	30451	30454
U 1358	30468	30492	30495	30499	30518	30521	30564	30567
U 1360	30606	30731	30735	30768	30830	30844	30924	30926
U 1368	30931	30968	30992	31108	31112	31123	31126	31129
U 1370	31163	31172	31175	31178	31182	31218	31227	31230
U 1378	31233	31235	31238	31241	31245	31313	31333	31337
U 1380	31349	31456	31479	31482	31514	31518	31595	31599
U 1388	31626	31630	31682	31685	31688	31692	31775	31780
U 1390	39822:	31853	31857	31860	31865	31871	31875	31881
U 1398	31884	31887	31893	31908	31911	31915	31920	31923
U 13A0	31926	31929	31933	31938	31941	31944	31948	32034
U 13A8	32037	32041	32044	32049	32096	32100	32112	32115
U 13B0	32119	32124	32131	32135	32152	32155	32159	32163
U 13B8	32172	32177	32203	32206	32240	32243	32277	32280
U 13C0	32379	32385	32442	32446	32463	32467	32482	32485
U 13C8	32498	32502	32511	32514	32518	32525	32586	32607
U 13D0	32611	32615	32671	32682	32702	32707	32727	32730
U 13D8	32733	32823	33011	33014	33019	33115	33120	33128

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 13E0	33131	33134	33138	33156	33164	33167	33172	33176
U 13E8	33182	33190	33198	33208	33215	33218	33221	33229
U 13F0	33232	33235	33245	15870:	15847:	15852:	15857:	15862:
U 13F8	16589:	16595:	16599:	16604:	16656:	16661:	16668:	16673:
U 1400	41954=	41210	41215	41959=	41964=	41381=	41225	41385=
U 1408	41351=	41355=	41228	42667=	41359=	41362=	41233	42670=
U 1410	41252	42711=	42716=	41301	42720=	41435=	41310	41439=
U 1418	41260=	41367	41265=	41271=	41323=	41329=	41333=	41337=
U 1420	41443=	41448=	41372	42038:	41414=	41420=	41425=	41429=
U 1428	41405	41306:	41970=	41974=	41409	42015:	42029=	42033=
U 1430	41457=	41461=	41465=	41452	41648=	41651=	41505	41568=
U 1438	41486=	41544	41490=	41494=	41669=	41673=	41591	41573=
U 1440	41525=	41595	41529=	41533=	41600=	41614	41604=	41608=
U 1448	41693=	41623	41698=	41703=	41763=	41628	41767=	41772=
U 1450	42049=	42053=	42057=	42061=	41657	42099=	42103=	42109=
U 1458	41688	42159:	41728	42164:	41746	42134=	41750	42138=
U 1460	42143=	42150=	41777	42154=	42376=	41782	42380=	42384=
U 1468	42435=	41787	42439=	42443=	42538=	41869	42542=	42546=
U 1470	42621=	41901	42626=	42631=	41709=	41713=	42686=	42689=
U 1478	43178=	41932	43182=	43186=	23985:	23989:	41719=	41722=
U 1480	42087=	42091=	42171:	42176:	43199=	42041	43204=	43209=
U 1488	43227=	43232=	43237=	42045	42352=	42357=	42412=	42415=
U 1490	42483=	42486=	42512=	42516=	42612=	42615=	42678=	42683=
U 1498	42703=	42707=	42821=	42824=	42831=	42834=	42847=	42850=
U 14A0	42967=	42970=	42977=	42980=	42993=	42996=	43109=	43112=
U 14A8	43118=	43121=	43132=	43135=	43140=	43143=	43169=	43172=
U 14B0	42083	42021:	41342:	41346:	43216=	43219=	42324	42259:
U 14B8	42114:	42119:	42125:	42129:	43243=	43246=	43252=	43255=
U 14C0	43261=	43264=	42389	42393	42448:	42451:	42489	42493
U 14C8	42508	42551	42604:	42607:	42652	42655	42658	42662
U 14D0	42695:	42698:	42815	42840	42855	42860	42866	42986
U 14D8	43001	43007	43104	43126	43148	43153	43159	43192
U 14E0	33563=	33567=	33572=	34380=	33496	33575=	33579=	34383=
U 14E8	33471=	33475=	33515=	33518=	36475=	36479=	36483=	36486=
U 14F0	33500	33619=	34437=	33624=	33739=	33742=	34441=	33629=
U 14F8	33820=	33823=	34495=	33611=	33828=	33831=	34498=	33615=
U 1500	33711=	33716=	33722=	33728=	33732=	33503	33535=	33538=
U 1508	33506	33639=	34566=	33643=	33859=	33863=	34570=	33669=
U 1510	33510	33894=	34275=	34280=	33899=	33904=	33908=	33912=
U 1518	33522	33650=	34600=	33654=	34129=	34133=	34604=	33673=
U 1520	39653=	34017=	34022=	34025=	39656=	34029=	34369=	34373=
U 1528	46986=	33660=	46990=	33664=	34168=	34172=	33525	33677=
U 1530	34137=	34141=	34147=	34150=	34154=	34157=	34693=	34697=
U 1538	33783=	33788=	33793=	33799=	33802=	33806=	33810=	33814=
U 1540	34245=	34249=	34256=	34260=	34264=	34268=	46911=	46915=
U 1548	33999=	34005=	33527	34008=	36492=	36496=	36500=	36504=
U 1550	34178=	34181=	34287=	34291=	34294=	34716=	33530	34720=
U 1558	33542	34068=	34072=	34076=	36559=	36563=	36567=	36570=
U 1560	34193=	34198=	34394=	34398=	34237=	34241=	33545	34401=
U 1568	34080=	34085=	34089=	34093=	36576=	36580=	36584=	36588=
U 1570	34447=	34451=	34455=	33549	34315=	34319=	34458=	34463=
U 1578	34539=	34542=	33553	33942=	36509=	36512=	33556	33945=
U 1580	39879:	39885:	39890:	39896:	39899:	39903:	47003=	47006=

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 1588	34355=	34359=	34363=	33559	36627=	36631=	33633	46373=
U 1590	39913:	39920:	39925:	39932:	39935:	39939:	36688=	36692=
U 1598	33777	36636=	33836	36639=	36697=	36701=	33917	46378=
U 15A0	39969:	39976:	39980:	39984:	39987:	39991:	36726=	36729=
U 15A8	36675=	36679=	36684=	33948	36752=	36755=	33960	46383=
U 15B0	34725=	34729=	34733=	34737=	34741=	34745=	34748=	34751=
U 15B8	34755=	34759=	34762=	33994	36773=	36777=	34012	46387=
U 15C0	40056=	40503=	40061=	40506=	40066=	40071=	40076=	40081=
U 15C8	36809=	36813=	34033	36782=	36849=	36853=	34038	36785=
U 15D0	34162	40528=	34187	40532=	40087=	40091=	34298	40096=
U 15D8	40484:	40489:	40494:	40498:	36858=	36861=	34301	46468=
U 15E0	40142=	40538=	40146=	40542=	40150=	40154=	40159=	40164=
U 15E8	36908=	36911=	34305	36896=	39726=	39731=	34310	36900=
U 15F0	40169=	40173=	34351	40180=	39740=	39745=	34388	40185=
U 15F8	40548=	40552=	40557=	40561=	40365=	40368=	34405	46472=
U 1600	40191=	40195=	34409	40201=	40715=	40719=	34432	40208=
U 1608	34489	40626=	34501	40630=	36966=	36970=	36974=	36978=
U 1610	40256=	40262=	40267=	34505	34510	46349=	40270=	46353=
U 1618	40902:	40906:	40910:	40914:	39802:	39806:	39810:	39813:
U 1620	40400=	40403=	40408=	40411=	40414=	40417=	40422=	40427=
U 1628	40930:	40934:	40938:	40942:	46145:	46148:	40035=	40039=
U 1630	46152=	46156=	46159=	34536	40432=	40436=	34544	40439=
U 1638	46171=	46176=	46179=	34547	40775=	40778=	40046=	40050=
U 1640	46529=	46534=	34550	46539=	46552=	46557=	34553	46560=
U 1648	46227=	46231=	34557	40213=	46243=	46247=	34561	40218=
U 1650	34595	46721=	34607	46725=	34610	46736=	34613	46740=
U 1658	46815=	46819=	46823=	34616	46251=	46254=	40377=	40380=
U 1660	46832=	46835=	46839=	34620	47101=	47105=	47109=	34624
U 1668	42581:	42584:	42674:	34658	46287=	46291=	40387=	40390=
U 1670	42404:	42407:	42566:	34661	47154=	47158=	47162=	34663
U 1678	47165=	47169=	47173=	34666	40568=	34670	40572=	40577=
U 1680	47197=	47201=	47204=	34673	40693:	40696:	40702:	40706:
U 1688	47211=	47215=	47219=	34676	46305=	46309=	34679	40709:
U 1690	47237=	47241=	47244=	34682	47251=	47255=	47259=	34685
U 1698	34688	47332=	47338=	47341=	40647=	40651=	34701	40657=
U 16A0	34704	47346=	34707	47349=	40756=	40759=	40763=	40767=
U 16A8	47356=	47361=	47366=	47371=	46313=	46316=	34710	40770=
U 16B0	47398=	47402=	47406=	34714	47423=	47426=	47431=	36488
U 16B8	46391=	46394=	46399=	46403=	40815=	40818=	40822=	40825=
U 16C0	46407=	46412=	46416=	46419=	46424=	46428=	46432=	46437=
U 16C8	46441=	46444=	46449=	46453=	40858=	36523	40863=	40866=
U 16D0	46457=	46462=	46475=	46478=	46484=	46488=	46492=	46496=
U 16D8	46544=	46547=	46569=	46573=	40874=	40878=	40882=	40886=
U 16E0	46576=	46580=	46583=	46587=	46590=	46593=	46595=	46598=
U 16E8	46601=	46605=	46608=	46612=	46615=	46619=	36527	47083=
U 16F0	46622=	46626=	46629=	46632=	46635=	46639=	46642=	46646=
U 16F8	46648=	46652=	46655=	46659=	46662=	46665=	46668=	46672=
U 1700	46675=	46679=	46682=	46686=	46689=	46693=	46696=	46716=
U 1708	46744=	46749=	46772=	46776=	46871=	46875=	46879=	46883=
U 1710	46886=	46890=	46893=	46895=	46922=	46926=	46952=	46955=
U 1718	46993=	46996=	47012=	47015=	47022=	47049=	47053=	47056=
U 1720	47070=	47073=	47087=	47091=	47094=	47098=	47112=	47136=
U 1728	5496:	5500:	42571:	42574:	47176=	47181=	47222=	47227=

Location / Line Number Index

Location	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
U 1730	47262=	47265=	47302=	47323=	47378=	47382=	47412=	47415=
U 1738	36572	36615	23993:	23997:	36619	36622	36643	36707
U 1740	36712	36715	36719	36723	36732	36736	36739	36742
U 1748	36745	36749	36759	36763	36767	36788	36791	36793
U 1750	36796	36799	36803	36818	36867	36871	36875	36878
U 1758	36882	36885	36888	36892	36905	36916	36919	36922
U 1760	36926	36929	36932	36935	36938	36942	39693	39755
U 1768	39760	39796:	39817:	39828:	40028	40042	40241	40245
U 1770	40250	40297	40300	40303	40306	40309	40313	38370:
U 1778	40316	40319	40321	40324	40327	40330	40333	38374:
U 1780	40337	40371	40383	40394	40441	40445	40509	40512
U 1788	40515	40518	40522	40581	40610	40622	40633	40636
U 1790	40639	40643	40723	40782	40870	46162	40687:	46166
U 1798	40751:	46183	40810:	40854:	46187	46201	46212	46215
U 17A0	46219	46235	46239	46276	46280	46295	46299	46301
U 17A8	46344	46357	46360	46364	46368	46523	46563	46566
U 17B0	46731	46763	46768	46781	46807	46810	46827	46898
U 17B8	46908	46919	46944	46947	46977	46980	46983	47000
U 17C0	47009	47019	47059	47062	47066	47139	47144	47147
U 17C8	47150	47184	47187	47190	47193	47208	47230	47233
U 17D0	47248	47268	47271	47274	47278	47281	47284	47326
U 17D8	47353	35655:	47375	35660:	47386	47389	35636:	35640:
U 17E0	5622:	5626:	18424=	18427=	18208	18388=	18227	18391=
U 17E8	18364=	18368=	18372=	18375=	18379=	18383=	18436=	18439=
U 17F0	42003:	42009:	41981:	41986:	39485:	39489:	39834:	41378:
U 17F8	36518:	37652:	39649:	35629:	35632:	47295:	47299:	37892:
U 1800 - 281F Unused								
U 2820						20788:		20792:
U 2828 - 28C7 Unused								
U 28C8	21682:	21686:	21690:	21693:			45002:	
U 28D0 - 2927 Unused								
U 2928				22008:	22012:			
U 2930			22017:	22021:	22025:	22028:		
U 2938 - 2A47 Unused								
U 2A48				40222:	40232:	40237:		40226:
U 2A50 - 2A77 Unused								
U 2A78			40605:	40597:	40586:	40600:	40590:	40594:
U 2A80 - 2B47 Unused								
U 2B48				37698:	37701:	37705:	37709:	37712:
U 2B50	37719:	37733:	37723:	37727:	37737:	37755:	37740:	37744:
U 2B58	37748:	37752:	37758:	37765:	37770:	37773:	37776:	37779:
U 2B60	37783:	37789:	37792:	37798:	37801:	37804:	37807:	37810:
U 2B68	37813:	37816:	37819:	37822:	37827:	37831:		
U 2B70 - 2BEF Unused								
U 2BF0							36152:	

```
; This page intentionally left blank.
```

C-code Microword Summary

	Words not in bounds
REV750	0
CHARTS	0
REGION	0
DEFIN	0
MACRO	0
INIT	0
CONSOL	0
INTLOG	0
FLOAT	0
VIELD	0
CONTRL	0
PCALL	0
MISQUE	0
CHAR	0
DECMAL	0
EDIT	0
MPRCHM	0
PRLDSV	0
IANDE	0
MM	0
CMODE	256
OSR	0
UVERFY	0
GHROM	0
FILLER	0
Used	256
Remaining	

Total microwords used in memory C: 256
Total microwords remaining in memory C: 0
Highest address used in memory C: OFF (hex)

; This page intentionally left blank.

D-code Microword Summary

; This page intentionally left blank.

I-code Microword Summary

	Words not in bounds
REV750	0
CHARTS	0
REGION	0
DEFIN	0
MACRO	0
INIT	0
CONSOL	0
INTLOG	92
FLOAT	47
YIELD	.
CONTRL	39
PCALL	5
MISQUE	16
CHAR	11
DECMAL	16
EDIT	1
MPRCHM	6
PRIDSV	4
IANDE	9
MM	0
CMODE	0
OSR	0
OVERFV	0
CHROM	0
FILTER	0

Used 255
Remaining

Total microwords used in memory [: 255
Total microwords remaining in memory [: 1
Highest address used in memory [: 0FF (hex)

```
; This page intentionally left blank.
```

; CMT098.MCX MICR' ' 1M(01) 28-NOV-83 16:30:35 CLOKX Rev 13.00, Clock rate = 160ns
;
0-code Microword Summary

	Words not in bounds
REV750	0
CHARTS	0
REGION	0
DEFIN	0
MACRO	0
INIT	0
CONSOL	0
INTLOG	97
FLOAT	47
VIELD	7
CONTRL	39
PCALL	3
MISQUE	16
CHAR	11
DECMAL	16
EDIT	1
MPRCHM	6
PRLDSV	4
IANDE	9
MM	0
CMODE	0
OSR	0
UVERFY	0
GHROM	0
FILLER	0

Used 256
Remaining

Total microwords used in memory 0: 256
Total microwords remaining in memory 0: 0
Highest address used in memory 0: 0FF (hex)

; This page intentionally left blank.

U-code Microword Summary

	ROW 1 0000-03FF	ROW 2 0400-07FF	ROW 3 0800-0BFF	ROW 4 0C00-0FFF	ROW 5 1000-13FF	ROW 6 1400-17FF	WCS 2000-23FF	PCS 2800-2BFF	Words not in bounds
REV750	0	0	0	0	0	0	0	0	0
CHARTS	0	0	0	0	0	0	0	0	0
REGION	0	0	0	0	0	0	0	0	0
DEFIN	0	0	0	0	0	0	0	0	0
MACRO	0	0	0	0	0	0	0	0	0
INIT	1	9	129	0	0	4	0	0	0
CONSOL	1	0	438	4	0	0	0	0	0
INTLOG	113	122	0	2	0	0	0	0	0
FLOAT	228	556	1	0	13	0	0	0	0
VIELD	77	0	0	0	0	14	0	0	0
CONTRL	64	84	0	0	0	0	0	0	0
PCALL	2	2	0	151	0	0	0	2	0
MISQUE	31	0	2	154	0	0	0	10	0
CHAR	28	0	436	0	0	4	0	0	0
DECMAL	53	5	9	111	1010	0	0	0	0
EDIT	5	0	2	0	0	206	0	0	0
MPRCHM	15	3	0	305	0	6	0	1	0
PRLDSV	6	0	1	0	0	99	0	0	0
IANDE	42	7	4	295	0	6	0	35	0
MM	15	0	0	0	1	184	0	10	0
CMODE	147	4	0	0	0	235	0	0	0
OSR	193	130	1	0	0	0	0	1	0
UVERFY	3	0	1	0	0	266	0	0	0
GHROM	0	98	0	0	0	0	0	0	0
FILLER	0	4	0	2	0	0	0	0	0
Used	1024	1024	1024	1024	1024	1024	0	59	0
Remaining	0	0	0	0	0	0	1024	965	

Total microwords used in memory U: 6203
Total microwords remaining in memory U: 1989
Highest address used in memory U: 2BF6 (hex)

; This page intentionally left blank.

; The files used in this assembly are:
REV750.MIC
CHARTS.MIC
REGION.MIC
DEFIN.MIC
MACRO.MIC
INIT.MIC
CONSOL.MIC
INTLOG.MIC
FLOAT.MIC
YIELD.MIC
CONTRL.MIC
PCALL.MIC
MISQUE.MIC
CHAR.MIC
DECMAL.MIC
EDIT.MIC
MPRCHM.MIC
PRLDSV.MIC
IANDE.MIC
MM.MIC
CMODE.MIC
OSR.MIC
UVERFY.MIC
GHROM.MIC
FILLER.MIC

21675 Validity failure in file MISQUE.
PUSH,NEXT/MM.PRB.WRITE.SIZ ; CHECK THAT TAIL CAN BE WRITTE

Pass 1 warnings: 0 Pass 2 warnings: 1
Pass 1 errors: 0 Pass 2 errors: 0

No. of micro words	6203	
No. extended	518	8.35%
No. over extended	0	
No. patches	0	
Fatal mul/div errs	0	

```
MICRO2  1M(01)  E 10
MICRO2  1M(01)  F 10
MICRO2  1M(01)  G 10
MICRO2  1M(01)  H 10
MICRO2  1M(01)  I 10
MICRO2  1M(01)  J 10
MICRO2  1M(01)  K 10
MICRO2  1M(01)  L 10
MICRO2  1M(01)  M 10
MICRO2  1M(01)  N 10
MICRO2  1M(01)  B 11
MICRO2  1M(01)  C 11
MICRO2  1M(01)  D 11
MICRO2  1M(01)  E 11
MICRO2  1M(01)  F 11
MICRO2  1M(01)  G 11
MICRO2  1M(01)  H 11
MICRO2  1M(01)  I 11
MICRO2  1M(01)  J 11
MICRO2  1M(01)  K 11
MICRO2  1M(01)  L 11
MICRO2  1M(01)  M 11
MICRO2  1M(01)  N 11
MICRO2  1M(01)  B 12
MICRO2  1M(01)  C 12
MICRO2  1M(01)  D 12
MICRO2  1M(01)  E 12
MICRO2  1M(01)  F 12
MICRO2  1M(01)  G 12
MICRO2  1M(01)  H 12
MICRO2  1M(01)  I 12
MICRO2  1M(01)  J 12
MICRO2  1M(01)  K 12
MICRO2  1M(01)  L 12
MICRO2  1M(01)  M 12
MICRO2  1M(01)  N 12
MICRO2  1M(01)  B 13
MICRO2  1M(01)  C 13
MICRO2  1M(01)  D 13
MICRO2  1M(01)  E 13
MICRO2  1M(01)  F 13
MICRO2  1M(01)  G 13
MICRO2  1M(01)  H 13
MICRO2  1M(01)  I 13
MICRO2  1M(01)  J 13
MICRO2  1M(01)  K 13
MICRO2  1M(01)  L 13
MICRO2  1M(01)  M 13
MICRO2  1M(01)  N 13
MICRO2  1M(01)  B 14
MICRO2  1M(01)  C 14
MICRO2  1M(01)  D 14
MICRO2  1M(01)  E 14
MICRO2  1M(01)  F 14
MICRO2  1M(01)  G 14
MICRO2  1M(01)  H 14
MICRO2  1M(01)  I 14
MICRO2  1M(01)  J 14
MICRO2  1M(01)  K 14
MICRO2  1M(01)  L 14
MICRO2  1M(01)  M 14
MICRO2  1M(01)  N 14
MICRO2  1M(01)  B 15
MICRO2  1M(01)  C 15
MICRO2  1M(01)  D 15
MICRO2  1M(01)  E 15
MICRO2  1M(01)  F 15
MICRO2  1M(01)  G 15
MICRO2  1M(01)  H 15
MICRO2  1M(01)  I 15
MICRO2  1M(01)  J 15
MICRO2  1M(01)  K 15
MICRO2  1M(01)  L 15
MICRO2  1M(01)  M 15
MICRO2  1M(01)  N 15
MICRO2  1M(01)  B 16
MICRO2  1M(01)  C 16
MICRO2  1M(01)  D 16
MICRO2  1M(01)  E 16
MICRO2  1M(01)  F 16
MICRO2  1M(01)  G 16
MICRO2  1M(01)  H 16
MICRO2  1M(01)  I 16
MICRO2  1M(01)  J 16
MICRO2  1M(01)  K 16
MICRO2  1M(01)  L 16
MICRO2  1M(01)  M 16
MICRO2  1M(01)  N 16
MICRO2  1M(01)  B 17
MICRO2  1M(01)  C 17
MICRO2  1M(01)  D 17
MICRO2  1M(01)  E 17
MICRO2  1M(01)  F 17
MICRO2  1M(01)  G 17
MICRO2  1M(01)  H 17
MICRO2  1M(01)  I 17
MICRO2  1M(01)  J 17
MICRO2  1M(01)  K 17
MICRO2  1M(01)  L 17
MICRO2  1M(01)  M 17
MICRO2  1M(01)  N 17
MICRO2  1M(01)  B 18
MICRO2  1M(01)  C 18
MICRO2  1M(01)  D 18
MICRO2  1M(01)  E 18
MICRO2  1M(01)  F 18
MICRO2  1M(01)  G 18
MICRO2  1M(01)  H 18
MICRO2  1M(01)  I 18
MICRO2  1M(01)  J 18
MICRO2  1M(01)  K 18
MICRO2  1M(01)  L 18
MICRO2  1M(01)  M 18
MICRO2  1M(01)  N 18
MICRO2  1M(01)  B 19
MICRO2  1M(01)  C 19
MICRO2  1M(01)  D 19
```

```
M 14                MICRO2  1M(01)
N 14                MICRO2  1M(01)
R 15                MICRO2  1M(01)
C 15                MICRO2  1M(01)
D 15                MICRO2  1M(01)
E 15                MICRO2  1M(01)
F 15                MICRO2  1M(01)
G 15                MICRO2  1M(01)
H 15                MICRO2  1M(01)
```