PERKIN-ELMER

OS/32
OPERATOR
REFERENCE MANUAL

TABLE OF CONTENTS

CHAPTERS (Continued)

CHAPTERS (Continued)

CHAPTERS (Continued)

CHAPTERS (Continued)

4   SYSTEM ERROR HANDLING

5   THE COMMAND SUBSTITUTION SYSTEM (CSS)

CHAPTERS (Continued)

FIGURES

TABLES

PREFACE

The OS/32 Operator Reference Manual is intended for OS/32 system operators.

Chapter 1 gives a general description of OS/32. Chapter 2 discusses the procedures for loading either the Perkin-Elmer supplied starter systems or user-generated systems. Chapter 3 defines the operator commands. OS/32 now supports the Perkin-Elmer Model 3200MPS System. Two new operator commands have been added to Chapter 3, and several existing operator commands have been modified to accommodate the multiple processor configurations of the Model 3200MPS System. Chapter 4 discusses error handling and maintenance utility programs. Chapter 5 discusses the command substitution system (CSS). Chapter 6 briefly describes the accounting facility used in monitoring system usage. Appendix A contains a summary of operator commands. Appendix B is a command message summary. Appendix C describes system messages. Appendix D contains system crash codes. Appendix E is a control summary for the new bidirectional input/output control (BIOC).

This manual is intended for use with the OS/32 R06.2 software release or higher. Additional material specifically related to the Model 3200MPS System has also been included. These Model 3200MPS System features are supported by the OS/32 R07.1 software release and higher. Throughout the text, these features are identified as applicable only to the Model 3200MPS System.

For further information on the contents of all Perkin-Elmer 32-bit manuals, see the 32-Bit Systems User Documentation Summary.

CHAPTER 1
INTRODUCTION TO COMPUTER OPERATIONS UNDER OS/32

1.1 INTRODUCTION

OS/32 is a comprehensive multi-tasking environment with minimal system overhead that is controlled from an interactive console. The OS/32 environment includes foreground tasks for combining and forming application systems, and background tasks for program development and general data processing.

1.2 MINIMUM HARDWARE CONFIGURATIONS

The minimum hardware configurations for OS/32 are:

- Perkin-Elmer 32-bit processor with 128kb of memory

- Relocation/protection hardware; memory access controller (MAC) or memory address translator (MAT)

- Display panel (for models 7/32 and 8/32 processors only)

- Universal clock

- Console device options:

  - Teletypes (TTYs) (Models 33 and 35)

  - Perkin-Elmer Carousel 15, 30, 35, or 300

  - Perkin-Elmer Model 550, 550B, 1100, 1200, 1250, or 1251 video display unit (VDU)

- Power fail/auto restart

- Magnetic media options:

  - Magnetic tape (9-track 800, 1600, 6250 bits per inch (bpi)

  - Disks (2.5Mb, 10Mb, 40Mb, MSM80, MSM300, MSM625)

  - Floppy disk

- Input/Output (I/O) device options:

  - Line printer (120, 180 characters per second (cps) 300, 600
    and 1000 lines per minute (lpm)

  - Card reader (400 or 1000 characters per minute (cpm)

  - Paper tape reader/punch

  - Card punch

- Data communications interfaces (PALS, PASLA, SSA, QSA, 2- and
  8-line communications multiplexor (COMM MUX), current loop
  communications multiplexor (CLCM)

- Special device options:

  - Loader storage unit (LSU)

  - Eight-line interrupt module

  - Video display (TTY interface or PASLA/PALS)

  - Digital multiplexor

  - Mini I/O system

  - Real-time analog system

- OS/32 can be configured to run without direct access devices.
  However, at least one disk is required to:

  - unpackage the software,

  - perform a system generation (sysgen), and

  - do program development.

1.3 THE OS/32 ENVIRONMENT

OS/32 is a general purpose multitasking, multi-environment
system. Basic OS/32 provides a background environment for
program development and debugging and a foreground environment
for prioritized real-time applications. Additional environments
are provided by these OS/32 products:

- Multi-terminal monitor (MTM) provides a secure multi-user,
  time-sharing environment for program development and
  debugging.

- Reliance is a complete online transaction processing software package for the commercial user of Perkin-Elmer 32-bit computers.

OS/32 also provides many facilities to allow complete application-oriented environments to be easily constructed.

1.3.1 The Operator in an OS/32 Environment

The OS/32 system is controlled by the system operator through a device called the system console. This device can be a TTY, VDU, Carousel, or a Model 550, 550B, 1100, 1200, 1250, or 1251 VDU. It has a special relationship to the system in that the system receives command input from the console and writes system messages to it. Tasks can log messages to the system console without reference to its device name. A comprehensive command set is provided to allow the system operator to control and interact with the various OS/32 environments. Normally, the operator does not interact with foreground tasks except when loading and initiating them, but the operator can monitor and control them, if necessary.

The background facilities offer the operator more control of jobs and job streams. These facilities are used for program development functions such as:

- assembling,

- compiling,

- linking, or

- performing a sysgen.

To use these functions, the operator must understand the task control functions available under OS/32 and the program's specific operational requirements. Through operator commands, the operator frequently interacts with the disk file system. These commands are detailed in Chapter 3.

The operator must understand the OS/32 command substitution system (CSS). These commands are detailed in Chapter 5.

MTM enables users to simultaneously share a 32-bit processor via terminals, which adds another dimension to the operator's activities.

1.4 SYSTEM CONSOLE USE

The system console is used to enter commands and receive status displays and error responses. Significant events such as disk failures, power fail/restore, and task terminations are logged to the console.

OS/32 provides, through the SET LOG command, an optional copy of all transactions to and from the console. The log may be directed to a hard copy device or to a file for later printing.

The system console can be assigned to tasks for ordinary I/O operations the same as any other device. If the system console is an ASR TTY or Carousel 35, only its keyboard/printer unit can be used for calling tasks; the reader/punch unit is reserved for system use.

1.4.1 Prompts

When the system operator is expected to enter data at the system console, a prompt is output. This prompt takes one of the following forms:

    *   Command request

    taskid>   Data request

    .CMDP>   Build request

The command request prompt (*) is output whenever the system is ready to accept another command.

The data request prompt (taskid>) is output whenever a task is attempting to perform a read request to the system console. The task identifier (taskid) of this prompt is the name of the task requesting data. For the background task, the taskid is .BG. The system operator should satisfy the data request as soon as possible, since system messages are held up until the data request is satisfied.

The build request prompt (.CMDP>) is output whenever the command processor task is requesting input. This occurs during the processing of a BUILD command.

1.4.2 BREAK Key

If a task is in the process of reading from or writing to the system console, the operator can interrupt by pressing the BREAK key (ESC on some devices) of the console device. This forces the system into command mode for the entry of one command line. After the command line has been accepted, the user I/O to the console is restarted. This process is transparent to the user task (u-task).

The BREAK key can also be used by the operator to terminate further system responses to a command. This is particularly useful in cases such as the EXAMINE and DISPLAY commands, where large quantities of data may be output to the system console.

1.5 GENERAL SYNTAX RULES

Multiple commands may appear on one line, but each one must be separated by a semicolon (;). Certain commands must appear last on a line or must be the only command on the line. These restrictions are discussed in the sections dealing with the individual commands. If the first character of any command input is an asterisk (*), the remainder of that line is considered to be a comment and is not executed. It is copied to the system log device if logging is active.

1.5.1 Statement Syntax Conventions

The following statement syntax conventions are used in all statement, command, and instruction formats.

Underlining points out the mnemonic of the entry and means that at least the underlined portion must be entered. If an entry is not underlined at all, the entire entry must be entered.

     PAUSE

Capital letters must be entered exactly as shown.

     DELETE fd$_1$ [,fd$_2$,...,fd$_n$]

Lowercase letters represent parameters or denote information provided by the user:

     CANCEL taskid

Punctuation must be entered exactly as shown.

Commas separate parameters and substitute for missing positional parameters:

$$\text{DELETE } fd_1\ [,fd_2,\ldots,fd_n]$$

Commas preceding braces inside brackets must be entered if one of the optional parameters is chosen:

$$\text{DISPLAY ERRORS} \left[, \left\{ \begin{array}{c} fd \\ \text{system console} \end{array} \right\} \right]$$

Commas inside of brackets must be entered if the optional parameter is chosen:

$$\text{BFILE } fd[,lu]$$

An ellipsis represents an indefinite number of parameters or a range of parameters.

BUILD fd
.
.
.
ENDB

Brackets represent an optional parameter.

CONTINUE [address]

Braces represent required parameters from which one must be chosen.

$$\text{CLOSE} \left\{ \begin{array}{c} lu_1\ [,lu_2,\ldots,lu_n] \\ \text{ALL} \end{array} \right\}$$

Shading represents a default option.

$$\text{DISPLAY DEVICES} \left[, \left\{ \begin{array}{c} \text{fd} \\ \hline \text{system console} \end{array} \right\} \right]$$

An equal sign associates a parameter with its keyword.

$$\text{MARK dev:} , \left\{ \begin{array}{l} \text{PARAMETER} \; [,\text{RECL=recl}] \; [,\text{SIZE=size}] \\ \text{OFF} \\ \text{ON} \; \left[, \left\{ \begin{array}{l} \text{PROTECTED} \\ \text{SYSTEM}[/\text{max acct}] \\ \text{RESTRICTED}[/\text{max acct}]=\text{actno} \end{array} \right\} \right] \end{array} \right\} \left[,\text{DIRECTORY} \; \left[= \left\{ \begin{array}{c} \text{bsize} \\ \hline 80 \end{array} \right\}_{\text{ALL}} \right] \left[/ \left\{ \begin{array}{c} \text{exp} \\ \hline 100 \end{array} \right\} \right] \right] \right\}$$

1.5.2 Uppercase and Lowercase Characters

All commands and parameters can be entered in either uppercase or lowercase. Parameters that are retained internally (such as task identifiers) are translated to uppercase. A subsequent display will show the uppercase version.

1.5.3 Decimal and Hexadecimal Numbers

The OS/32 commands use decimal, rather than hexadecimal, numbers for most numeric operands. A major exception is addresses, which are expressed in hexadecimal. Numeric operands are always integers except for the SET SYS and TCOM commands, and the segment size increment field of the LOAD command where the decimal point is permissible. Leading zeros can be omitted in numeric operands, whether decimal or hexadecimal.

1.5.4 Task Identifiers

Task identifiers must consist of one to eight alphanumeric characters; the first character must be alphabetic. Valid task identifiers are:

    TASK3

    MAX

    X

    T997XY25

Examples of invalid task identifiers are:

34TASK	first character is not alphabetic
T43.2	contains a nonalphanumeric character
TASK12345	contains more than eight characters

The background task has the special identifier .BG.

1.5.5 File Descriptors (fds)

File descriptors, abbreviated as fd, are entered in a standard format.

Format:

$$\left[\begin{Bmatrix} \text{voln:} \\ \text{dev:} \end{Bmatrix}\right] \quad [\text{filename}] \quad [.[\text{ext}]] \quad [/\text{actno}]$$

Parameters:

voln: is the name of a disk volume. It may be from one to four characters long. The first character must be alphabetic and the remaining, alphanumeric. If voln: is not specified, the default is:

- TEMP volume for temporary files,

- SPOOL volume for spool files, and

- SYSTEM volume for all other files.

dev: is a 1- to 4-character device name. The first character must be alphabetic and the remaining, alphanumeric.

filename is the name of a file and is from one to eight characters long. The first character must be alphabetic and the remaining, alphanumeric. If a filename is specified when a device mnemonic is specified as dev, the filename is ignored.

.ext   is the name of the extension and consists of
                from one to three alphanumeric characters. If
                .ext is not specified, a default extension
                corresponding to the appropriate command is
                appended. If the period (.) is specified
                with no extension, the default is blanks.

actno   is a decimal number ranging from 0 through
                65,535 specifying the account number
                associated with the file. Account numbers 1
                through 65,535 (excluding 255) are used by
                MTM. Account number 255 is reserved for the
                authorized user utility. Account number 0 is
                used for system files and is the default for
                all operator commands.

Examples:

PACK:CAL.TSK/0   PACK: is the volume name, CAL is
                           the filename, .TSK is the extension
                           name, and 0 is the account number.

CONV.CAL   CONV is the filename, and .CAL is
                           the extension name with a default
                           account number on the default
                           volume.

CAL   CAL is the filename with a default
                           extension, default account number,
                           and default volume.

M300:MAR   M300: is the volume name, and MAR
                           is the filename with a default
                           extension and default account
                           number.

CARD: CARD: is the device mnemonic.

1.6 ERROR RESPONSE

If a command is not acceptable to the system or an error
condition is detected while processing a command, a message is
displayed on the system console. The general format of the
message is:

    xxxx=ERR   TYPE=yyyy   POS=zzzz

Where:

    xxxx   is an error descriptor of up to four characters (such as MNEM, ALLO, IO, etc.). Error descriptions are defined in each command description.

    yyyy   indicates the type of error encountered and is output only when an I/O error, supervisor call (SVC) 6, or SVC 7 file handler error is encountered.

    zzzz   represents the last command item processed. This field is most useful when multiple commands are entered on one line.

See Appendix B for a complete list of messages.

The error response to an unrecognized command is:

    MNEM-ERR

All commands following an erroneous command on a command line are ignored if the command line is entered from the system console. If the command line is entered from a CSS, all commands are handled as if they were on individual command lines.

If the particular command cannot be accepted because of the state of the system, this message is displayed:

    SEQ-ERR

Such restrictions are discussed in the explanations of the particular commands in Chapter 3.

CHAPTER 2
LOADING OS/32

2.1 INTRODUCTION

OS/32 is stored in object or image format on one of these auxiliary devices:

- Paper tape

- Tape cassette

- Floppy disk

- Magnetic tape

- Disk device

OS/32 can be loaded from an auxiliary device into memory by one of these loaders:

- Relocating (REL) loader, if OS/32 is in object format stored on paper tape or magnetic media

- Direct access bootstrap (BOOT) loader, if OS/32 is in image format stored on magnetic tape or disk device and is to be loaded into a Model 7/32 or 8/32 system

- Loader storage unit (LSU), if OS/32 is in image format stored on magnetic tape or disk device

These loaders are described in two sections. The first section describes the loading procedures used for the Models 7/32 and 8/32 systems which are entered through the display panel. The second section describes the loading procedures for the Perkin-Elmer Series 3200 processors, which are entered through the system control terminal (with device address X'10').

2.1.1 Starter Systems

All starter systems can be loaded from a magnetic tape device by the REL loader (object format only), BOOTLOADER, or the LSU. OS/32 can also be loaded from a disk device by the BOOTLOADER or LSU.

2.2 OS/32 BOOTSTRAP PUNCHER FOR PERKIN-ELMER MODELS 7/32 AND 8/32 PROCESSORS

The OS/32 bootstrap puncher converts the object versions of the 32-bit REL loader and the OS/32 bulk storage bootstrap loader into 50-sequence loadable form.

2.2.1 Starting the Bootstrap Puncher

1.   Mount the media containing the bootstrap puncher.

2.   Load the bootstrap puncher (BTPCH32).

       LOAD

3.   Assign the input device or file from which the REL or BOOTLOADER is read.

       ASSIGN 1,fd

4.   Assign the output device to which the REL or BOOTLOADER is written.

       ASSIGN 2,fd

5.   Start the bootstrap puncher.

       START

The logical unit (lu) assignments do not have to be made using the ASSIGN command.   The assignments can be made via the START command.

Format:

       START,fd$_1$,fd$_2$

Where:

       fd$_1$ is the file descriptor of the input device.

       fd$_2$ is the file descriptor of the output device.

Example:

```
LO BTPCH32
AS 1,BOOT32.OBJ
AS 2,MAG2:
ST
OS/32 BOOT PUNCHER 03-00
BGG      -END OF TASK CODE=   0     CPUTIME=0.026/0.026
LO COPY32
ST
OS/32 COPY R01-00
OPT BIN,NOT,NOPS
COPY OS32LAB.006,MAG2:
END OF MEDIUM - 665 RECORD(S) COPIED
COPY COMPLETE - 1 FILE(S) COPIED
REW 0
END
BGG      -END OF TASK CODE=   0     CPUTIME=1.687/0.071
```

This example places the BOOTLOADER on the front end of MAG2: followed by a copy of an OS/32 image file.

2.2.2 Messages for Bootstrap Puncher

All messages are output to the system console in the following formats:

Messages:

ASGN-ERR TYPE=xxxx LU=yy

> indicates that the variable xxxx is a type of input/output (I/O) error and file access error. yy is the lu where the error occurred.

8100 ERROR:LU=xx

> indicates that the variable xx is the lu not assigned.

ILLEGAL ASSIGNMENT ON LUx

> indicates that the variable x is the lu where the error occurred.

Table 2-1 defines the error types.

TABLE 2-1 ERROR TYPES

ERROR TYPE	xxxx	DEFINITION
I/O	ILLU	Illegal or unassigned lu
	PRTY	Parity or recoverable error
	UNRV	Unrecoverable error
	EOF	End of file
	EOM	End of medium
	DU	Device unavailable
	FUNC	Invalid function or device
File access	FUNC	Illegal function
	VOL	No such volume/device
	LU	Illegal lu
	NAME	Mismatched filename.ext
	SIZE	Erroneous record length or size
	PROT	Mismatched protection key
	PRIV	Mismatched access privileges
	BUFF	Unable to obtain file control block (FCB)
	TYPE	Nondirect access device

2.3 LOADING OS/32 FOR MODELS 7/32 AND 8/32 SYSTEMS

The following sections describe the procedures for loading the desired loader and OS/32 through the display panel.

2.3.1 Loading OS/32 Using the Relocating (REL) Loader

To load OS/32 object format, follow this procedure:

1. Mount the media containing the REL loader and position it at load point.

2. Enter the 50-sequence:

        DTA-50-ADD
        DTA-D500-WRT
        DTA-00CF-WRT
        DTA-4300-WRT
        DTA-0080-WRT

3. Enter into memory starting at location X'78' the device number and output command of the device containing the REL loader:

DTA-78-ADD	Address location X'78'

$$
DTA-\begin{cases} 0294 \\ 0399 \\ 1399 \\ 45A1 \\ 55A1 \\ 85A1 \\ C5A1 \end{cases}-WRT
$$

0294	Teletype reader
0399	High-speed paper tape reader
1399	High-speed reader/punch
45A1	Tape cassette (deck 0)
55A1	Tape cassette (deck 1)
85A1	Magnetic tape (800 bpi)
C5A1	Magnetic tape (1600 bpi)

4. Clear the program status word (PSW) by entering:

DTA-0-FN-1

5. Initialize machine malfunction pointers by entering:

DTA-84-ADD	Address location X'84'
DTA-5000-WRT	Set pointer to 5000
DTA-6000-WRT	Set pointer to 6000

6. If the processor is equipped with an LSU, the LSU must be off.

7. Initialize the processor and start executing at location X'50' by entering:

INI-DTA-50-ADD-RUN

8. Display the REL loader starting address by entering:

FUN-LOC

9. If OS/32 is not on the same media as the REL loader, mount the media containing OS/32 and enter into location X'78' the device number and output command of the device containing OS/32. See step 3.

10. Specify the number of filemarks to skip and execute the REL loader to load OS/32 by entering:

```
DTA-rel addr-ADD
DTA-00nn-RUN
```

Where:

nn   is the number of filemarks

11. When loading is completed, the OS/32 id and license or contract are displayed on the system console:

```
OS/32nn-uu.vvvvvvvv    LICENSE x-xxxx-xxxx-x
```

Where:

nn   is a decimal number indicating the revision level.

uu   is a decimal number indicating the update level.

vvvvvvvv   is an 8-character alphanumeric string indicating the user version number.

x-xxxx-xxxx-x

            is a series of alphanumeric characters identifying the license or contract of the system.

2.3.2 Relocating (REL) Loader Error Check Procedure and Error Codes

If the WAIT light does not come on, the OS/32 revision id is not displayed on the system console, or zeros are not displayed on the display panel, follow this error check procedure:

1. Ensure that the device containing the REL loader is turned on and is online to the system.

2. Ensure that the device containing the REL loader is correctly positioned.

3. Ensure that the data entered in the appropriate locations is correct.

4. Load a different copy of the REL loader.

5. If the error is not detected after performing steps 1 through 4, a hardware error might exist. Check the display panel for the REL loader error codes (Table 2-2) or system crash codes (Appendix D).

TABLE 2-2   REL LOADER ERROR CODES

ERROR CODE	MEANING
0000	The load terminated normally.
0001	A checksum or parity error occurred. Press RUN to reread.
0002	A sequence error occurred; retry load. OS/32 may be defective or incomplete.
0003	An attempt to load over the loader occurred; retry.
0004	An REF-DEF chain loop occurred; retry.
0005	The device is unavailable or is not positioned to the correct file. Check the device address.
0006	Selector channel (SELCH) specified is incorrect. Check device and SELCH address.
0007	End of file (EOF) was encountered; retry load.
FFnn	A load error occurred because an improper loader item nn was detected. Correct the control item and retry.

2.3.3 Loading OS/32 Using the BOOTLOADER

To load OS/32 image format, follow this procedure:

1. Mount the media containing the BOOTLOADER (see Section 2.2) and position it at load point.

2. Mount the media containing OS/32 and wait until the READY light comes on.

3. Enter the 50-sequence:

        DTA-50-ADD   Address location X'50'
        DTA-D500-WRT
        DTA-00CF-WRT
        DTA-4300-WRT
        DTA-0080-WRT

4. Enter into memory starting at location X'78' the device number and output command of the device containing the BOOTLOADER:

        DTA-78-ADD   Address location X'78'

⎧ 0294 ⎫	Teletype
| 0399 |	High-speed paper tape reader
| 1399 |	High-speed reader/punch
DTA-⎨ 45A1 ⎬-WRT	Tape cassette (deck 0)
| 55A1 |	Tape cassette (deck 1)
| 85A1 |	Magnetic tape (800 bpi)
⎩ C5A1 ⎭	Magnetic tape (1600 bpi)

5. Enter into memory starting at location X'7A' the device number and device code of the device containing OS/32:

        DTA-7A-ADD   Address location X'7A'

⎧ C631 ⎫	2.5Mb disk
| C633 |	10Mb removable disk
| C732 |	10Mb fixed disk
| FC34 |	40Mb disk
DTA-⎨ FC35 ⎬-WRT	MSM80 disk
| FC36 |	MSM300 disk
| C137 |	Floppy disk
| 8540 |	Magnetic tape (800 bpi)
⎩ C541 ⎭	Magnetic tape (1600 bpi)

6. Enter into memory starting at location X'7C' the controller and SELCH addresses of the device containing OS/32:

DTA-7C-ADD   Address location X'7C'

```
      ⎧B6FO⎫
      ⎪B6FO⎪
      ⎪FBFO⎪
DTA-  ⎨FBFO⎬ -WRT
      ⎪FBFO⎪
      ⎪nn00⎪
      ⎩nnFO⎭
```

2.5Mb disk
10Mb disk
40Mb disk
MSM80 disk
MSM300 disk
Floppy disk (nn=drive number)
Magnetic tape (controller included) nn=driver number

7. If loading from disk, enter into memory at location X'7E' the extension field of the OS/32 file descriptor (fd) (X'000'-X'FFF'):

DTA-ext-WRT

8. Initialize machine malfunction pointers by entering:

DTA-84-ADD   Address location X'84'
DTA-5000-WRT   Set pointer to 5000
DTA-6000-WRT   Set pointer to 6000

9. If loading from disk, disable the write protect switch on the disk drive.

10. Initialize the processor and start executing at location X'50' by entering:

INI-DTA-50-ADD-RUN

11. When loading is completed, the OS/32 id and license or contract are displayed on the system console:

OS/32nn-uu.vvvvvvvv   LICENSE x-xxxx-xxxx-x

Where:

nn   is a decimal number indicating the revision level.

uu   is a decimal number indicating the update level.

vvvvvvvv   is an 8-character alphanumeric string indicating the user version number.

x-xxxx-xxxx-x   is a series of alphanumeric characters identifying the license or contract of the system.

2.3.4 BOOTLOADER Error Check Procedure and Error Codes

If the OS/32 id is not displayed on the system console within several minutes after initialization or zeros are not displayed on the display panel, follow these eight steps.

1. Ensure that the BOOTLOADER device is turned on and is online to the system.

2. Ensure that the disk's READY light is on.

3. Ensure that locations X'7A' through X'7D' contain a device address, device code, controller address, and SELCH address corresponding to the current hardware configuration.

4. Ensure that the system console is powered on and is online to the system.

5. Check the PSW status for a nonzero value.

6. If the error is not detected after performing steps 1 through 5, a hardware error may exist. Check the display panel for a BOOTLOADER error code (Table 2-3) or a system crash code (Appendix D).

7. Ensure that the console device configured in the system is correct for the specific hardware configuration.

8. Load a different copy of the BOOTLOADER.

TABLE 2-3 BOOTLOADER ERROR CODES

ERROR CODE	MEANING
1	No contiguous file having a name starting with the characters OS32 was found, or no OS32xxxx filename with the correct extension was found.
2	Status error for a 2.5Mb or 10Mb disk (device unavailable) occurred.
3	Status error for a 2.5Mb or 10Mb disk (seek incomplete, not ready) occurred.
4	Status error for a 40Mb, MSM80, or MSM300 disk (unsafe, not ready, device unavailable) occurred.
5	Status error for a 40Mb, MSM80, or MSM300 disk (unsafe, seek incomplete, or not ready) occurred.
6	Data transfer error occurred; for example, data parity error.
7	Memory access controller (MAC) test fault occurred.
8	Memory test fault.
9	Device or SELCH address error. Check device and SELCH addresses.

2.3.5 Loading OS/32 Using the Loader Storage Unit (LSU)

To load OS/32 from a contiguous disk file or tape using the LSU, follow this procedure:

1. Turn the LSU on/off switch to on.

2. Mount the disk volume or tape containing OS/32 and wait for the READY light to come on.

3. Enter into memory starting at location X'7A' the device address and device code of the device containing OS/32:

DTA-7A-ADD   Address location X'7A'

$$DTA \begin{cases} C631 \\ C633 \\ C732 \\ FC34 \\ FC35 \\ FC36 \\ C137 \\ 8540 \\ C541 \end{cases} WRT$$

2.5Mb disk
10Mb removable disk
10Mb fixed disk
40Mb disk
MSM80 disk
MSM300 disk
Floppy disk
Magnetic tape (800 bpi)
Magnetic tape (1600 bpi)

4. Enter into memory starting at location X'7C' the disk controller and SELCH addresses of the device containing OS/32:

$$DTA \begin{cases} B6F0 \\ B6F0 \\ FBF0 \\ FBF0 \\ FBF0 \\ nn00 \\ nnF0 \end{cases} WRT$$

2.5Mb disk
5Mb disk
40Mb disk
MSM80 disk
MSM300 disk
Floppy disk (n=drive number)
Magnetic tape (controller included) nn=drive number

5. If loading from disk, enter into memory starting at location X'7E' the extension field of the OS/32 fd (X'000' to X'FFF'):

DTA-ext-WRT

6. If loading from disk, disable the hardware write protect switch.

7. Initialize the processor and load OS/32 by pressing the INI key.

8. After the LSU loads OS/32 into memory, the OS/32 id and license or contract are displayed on the system console:

OS/32nn-uu.vvvvvvvv   LICENSE x-xxxx-xxxx-x

Where:

OS/32   is the standard OS/32 fd.

nn   is a decimal number indicating the revision level.

uu   is a decimal number indicating the update level.

vvvvvvvv   is an 8-character alphanumeric string indicating the user version number.

x-xxxx-xxxx-x   is a series of alphanumeric characters identifying the license or the system.

9. Turn the LSU ON/OFF switch to OFF.

2.3.6 Loader Storage Unit (LSU) Error Check Procedure and Error Codes

If the OS/32 id is not displayed on the system console within several minutes after initialization or zeros are not displayed on the display panel, follow this error check procedure:

1. Ensure that the LSU on/off switch is on.

2. Ensure that the READY light on the disk drive containing OS/32 is on.

3. Ensure that locations X'7A' through X'7D' contain a device address, output command, controller address, and SELCH address corresponding to the current hardware configuration.

4. Ensure that locations X'7E' and X'7F' contain the correct OS/32 fd extension.

5. Ensure that the system console is powered on and is online to the system.

6. Check the PSW status for a nonzero value.

7. If the error is not detected after performing steps 1 through 6, a hardware error may exist. Check the display panel for an LSU error code (Table 2-4) or a system crash code (Appendix D).

8. If a system failure code is not displayed on the display panel, check the hardware configuration for the system console device.

TABLE 2-4 LSU ERROR CODES

ERROR CODE	MEANING
1	No contiguous file having a name starting with the characters OS32 was found, or no OS32xxxx filename with the correct extension was found.
2	A 2.5Mb or 10Mb disk is unavailable.
3	A 2.5Mb or 10Mb disk is unsafe, not ready, or unavailable.
4	A 40Mb, MSM80, or MSM300 disk is unsafe, not ready, or unavailable.
5	A 40Mb, MSM80, or MSM300 disk is unsafe, not ready, or seek incomplete.
6	Data transfer error occurred.
7	The size of OS/32 is larger than the memory in which it is to be located.

2.4 LOADING OS/32 FOR PERKIN-ELMER SERIES 3200 SYSTEMS

The starter systems or OS/32 can be loaded from a magnetic tape or contiguous disk file by the LSU.

2.4.1 Loading OS/32 Using the Relocating (REL) Loader

To load OS/32 object format, follow this procedure:

1. Mount the media containing the REL loader and position it at load point.

2. Turn key to ON. Depress HALT/RUN to obtain prompt.

3. Enter the 50-sequence:

```
@50
=D500
=00CF
=4300
=0080
```

4. Enter into memory starting at location X'78' the device number and output command of the device containing the REL loader:

      @78   Address location X'78'

       =0294   Teletype reader
       =0399   High-speed paper tape reader
       =1399   High-speed reader/punch
       =45A1   Tape cassette (deck 0)
       =55A1   Tape cassette (deck 1)
       =85A1   Magnetic tape (800 bpi)
       =C5A1   Magnetic tape (1600 bpi)

5. Clear the PSW by entering:

      P
      =0

6. Initialize machine malfunction pointers by entering:

      @84   Address location X'84'

      =5000   Set pointer to 5000
      =6000   Set pointer to 6000

7. If the processor is equipped with an LSU, the LSU must be off.

8. Initialize the processor and start executing at location X'50' by entering:

      @50
      <

9. The REL loader starting address is displayed.

10. If OS/32 is not on the same media as the REL loader, load the media containing OS/32 and enter into location X'78' the device number and output command of the device containing OS/32. See step 3.

11. If the device containing OS/32 is magnetic tape, the SELCH address must be specified in location X'7D'. If the device containing OS/32 is not a tape, 0 must be specified in location X'7D'.

```
@7C                          Address location X'7C'

=00F0                        SELCH X'F0'
=00F1                        SELCH X'F1'
=0000                        No SELCH
```

12. Specify the number of filemarks to skip, and execute the REL loader to load OS/32 by entering:

```
@7E                          Address location X'7E'

=00nn                        nn=number of filemarks

@rel adr                     Address REL loader starting
                             address
<
```

13. When loading is completed, the OS/32 id and license or contract are displayed on the system console:

```
OS/32nn-uu.vvvvvvvv    LICENSE x-xxxx-xxxx-x
```

Where:

nn   is a decimal number indicating the revision level.

uu   is a decimal number indicating the update level.

vvvvvvvv   is an 8-character alphanumeric string indicating the user version number.

x-xxxx-xxxx-x   is a series of alphanumeric characters identifying the license or contract of the system.

2.4.2 Relocating (REL) Loader Error Check Procedure and Error Codes

If the WAIT light does not come on and the OS/32 revision id is not displayed on the system console, follow this error check procedure:

1. Ensure that the device containing the REL loader is turned on and is online to the system.

2. Ensure that the device containing the REL loader is correctly positioned.

3. Ensure that the data entered in the appropriate locations is correct.

4. Load a different copy of the REL loader.

5. If the error is not detected after performing steps 1 through 4, a hardware error might exist. Check the system console for the REL loader messages (Table 2-5) or system crash codes (Appendix D).

TABLE 2-5 REL LOADER MESSAGES

MESSAGE	MEANING
NORMAL END	The load terminated normally.
CKSM or PARITY ERR	A checksum error occurred. Press RUN to reread.
SEQ ERR	A sequence error occurred; retry load. OS/32 may be defective or incomplete.
ADDR ERR	An attempt to load over the loader occurred; retry.
REF-DEF ERR	An REF-DEF chain loop occurred; retry.
DEV ERR	Device is unavailable or is not assigned to the correct file. Check the device address.
CONFIG ERR	SELCH specified is incorrect. Check device address.
EOF ERR	End of file was encountered; retry load.
ILL LOADER ITEM ERR	A load error occurred because an improper control item nn was detected. Correct the control item and retry.

2.4.3 Loading OS/32 Using the 2kb Loader Storage Unit (LSU)

To load OS/32 image format from disk or magnetic tape using the 2kb LSU, follow this procedure:

1. Mount the media containing OS/32. If OS/32 is on a disk device, press the START switch and wait for the READY light to come on. If OS/32 is on magnetic tape, position the tape at load point.

2. Turn the LOCK/ON/STANDBY key to the ON position.

3. Press the initial power load (IPL) ENABLE/DISABLE switch to ENABLE.

4. Initialize and load the 2kb LSU BOOTLOADER program by pressing the INIT switch.

After the LSU program is initialized and loaded, the following is displayed on the system console:

```
3200 LSU LOADER Rnn-uu
DEVS
MG85
MGC5
DS5R
DS5F
DS67
D256
FLPY
OTHR
```

Where:

nn	is a decimal number indicating the revision level.
uu	is a decimal number indicating the update level.

After the device names are displayed on the system console, the LSU responds with the following prompts. All responses to prompts are lowercase to indicate that they are specified by the user. If an error is made when entering a response, press the BREAK key to discontinue the response and restart the loader. This causes the LSU id, followed by the device names, to be displayed on the system console again.

DEVICE=xxxx

Where:

xxxx   is a 1- to 4-character string indicating the
                    name of the device containing OS/32. If the
                    user-specified device mnemonic was not
                    displayed on the system console when the LSU
                    was initialized, enter the letters OTHR. See
                    Table 2-6 for a list of device descriptions.

If a magnetic tape device is specified as the response, the tape
is automatically rewound, and this prompt is displayed on the
system console:

FILEMARKS=xxx

Where:

xxx   is a 1- to 3-digit decimal number from 0
                    through 255 indicating the number of filemarks
                    to skip to reach the OS/32 file. When the
                    number of filemarks is entered, followed by a
                    carriage return, the LSU loads OS/32 from the
                    file beginning after the last filemark
                    skipped.

If a disk device is specified as the response, this prompt is
displayed on the system console with the volume name of the disk:

VOL=vvvv,FILE=filename.ext

Where:

vvvv   is a 1- to 4-character volume name of the disk
                    containing OS/32. The BOOTLOADER program
                    reads and displays the existing volume name to
                    the system console.

filename.ext   is the OS/32 filename and extension. After
                    the OS/32 filename and extension are entered,
                    followed by a carriage return, the LSU loads
                    OS/32.

If the letters OTHR are specified in response to the DEVICE=
prompt, these prompts are displayed on the system console one at
a time:

    DEV#=nnn

Where:

    nnn   is a 1- to 3-digit hexadecimal number
                      indicating the device address of the device
                      containing OS/32.

    CODE=cc

Where:

    cc   is a 2-digit hexadecimal number indicating the
                      device code of the device containing OS/32.
                      See Table 2-6 for a list of device codes.

    CTLR=mmm

Where:

    mmm   represents a 1- to 3-digit hexadecimal number
                      indicating the controller address of the disk
                      device containing OS/32. This prompt is not
                      displayed if the device code specified a
                      magnetic tape device.

    SLCH=sss

Where:

    sss   represents a 1- to 3-digit hexadecimal number
                      indicating the SELCH address of the magnetic
                      tape or disk device containing OS/32. This
                      prompt is not displayed if the device code
                      specified a floppy disk device.

    DRV#=n

Where:

n   is a decimal number indicating the spindle
                              containing the floppy disk from which the
                              OS/32 is to be loaded. This prompt is
                              displayed only if the device code indicated a
                              floppy disk device.

TABLE 2-6   PERKIN-ELMER STANDARD CONFIGURATIONS SUPPORTED
             BY LSU

DEVICE NAME	DEVICE DESCRIPTION	DEVICE ADDRESS	DEVICE CODE	CON- TROLLER ADDRESS	SELCH ADDRESS
MG85	800 bpi mag tape	85	40	N/A	FO
MG85	1600 bpi mag tape	C5	41	N/A	FO
DS5F	5Mb disk - fixed	C7	32	B6	FO
DS5R	5Mb disk - removable	C6	33	B6	FO
DS67	67Mb disk	FC	35	FB	FO
D256	256Mb disk	FC	36	FB	FO
FLPY	Floppy disk	C1	37	0	N/A

2.4.3.1 2kb Loader Storage Unit (LSU) Messages

FILE NOT FOUND

    indicates that the user-specified fd does not exist or is not
    contiguous.

IOERROR CNFG

    indicates that an I/O error occurred because the hardware
    configuration does not correspond with that specified by the
    user.

IOERROR DU

indicates that the user-specified device is unavailable.

IOERROR UNRE

indicates that the I/O error that occurred is unrecoverable.

MEMTST ERR nnnnnn

indicates that an error occurred during a memory test before
OS/32 was loaded. The faulting memory location is at nnnnnn.
Load an OS/32 with a size 2652 bytes less than location
nnnnnn to allow more room for the loader; otherwise, repair
memory.

2.4.4 Loading OS/32 Using the 8kb Loader Storage Unit (LSU)

To load OS/32 image format from disk or magnetic tape using the
8kb LSU BOOTLOADER program, follow this procedure:

1. Mount the media containing OS/32. If OS/32 is on a disk
    device, press the START switch and wait for the READY light
    to come on. If OS/32 is on magnetic tape, position the tape
    at load point.

2. Turn the LOCK/ON/STANDBY key to the ON position.

3. Place the IPL ENABLE/DISABLE switch in the ENABLE position.

4. Initialize and load the 8kb LSU BOOTLOADER program by
    pressing the INIT switch.

After the program is initialized and loaded, the following is
displayed on the system console.

```
BASIC TEST COMPLETE
3200 8KB LSU BOOTLOADER R01
DEVICE
MG85
MGC5
MG62
DS5R
DS5F
DS67
D256
MM68
MM01
MM67
C13R
C13F
C40F
C67F
FLPY
OTHR
DEVICE=
```

After the LSU program is loaded, processor capabilities are |
tested by the basic confidence test. If this test passes, the
following message is displayed:

    BASIC TEST COMPLETE

If this test fails, a message is displayed. See Section 2.4.4.2.
The basic confidence test also performs a checksum on the data
read from the 8kb LSU's privileged read-only memories (PROMs).
If the checksum calculated is incorrect, the following message is
displayed and the program is terminated:

    CHECKSUM ERROR

If a checksum error occurs, execute test 1 of the LSU support
program.   Test 1 tests the contents of the PROMs for accuracy
against an error-free BOOTLOADER program. If the contents of the
PROMs are correct, reinitialize the 8kb LSU program.   If a |
checksum error reoccurs, the problem might be hardware related.

After the basic confidence test message is displayed, the
following program id is displayed.

    3200 8KB LSU BOOTLOADER Rnn-uu

Where:

nn   is a decimal number indicating the revision level.

uu   is a decimal number indicating the update level.

After the LSU id is displayed on the system console, the operating system bootable device menu is shown. See Table 2-7 for a list of Perkin-Elmer 8kb LSU-supported device names, descriptions, and configurations.

TABLE 2-7 8KB LSU SUPPORTED DEVICE NAMES AND CONFIGURATIONS

DEVICE NAME	DEVICE DESCRIPTION	DEVICE ADDRESS	DEVICE CODE	CON-TROLLER ADDRESS	SELCH ADDRESS
MG85	800 bpi mag tape	85	40	N/A	F0
MGC5	1600 bpi mag tape	C5	41	N/A	F0
MG62	6250 bpi mag tape	85	44,45, or 46	N/A	F0
DS5R	5Mb disk - removable	C6	33	B6	F0
DS5F	5Mb disk - fixed	C7	32	B6	F0
DS67	67Mb disk	FC	35	FB	F0
D256	256Mb disk	FC	36	FB	F0
MM68	68.7Mb disk - MMD fixed and head per track	FC	38	FB	F0
MM01	1.6 Mb disk - MMD head per track	FC	39	FB	F0
MM67	67.2Mb disk - MMD fixed	FC	3A	FB	F0
C13R	13.5Mb disk - CMD removable	FC	3B	FB	F0
C13F	13.5Mb disk - CMD fixed	FC	3C	FB	F0
C40F	40.4Mb disk - CMD fixed	FC	3D	FB	F0
C67F	67.3Mb disk - CMD fixed	FC	3E	FB	F0
FLPY	Floppy disk	C1	37	N/A	N/A

N/A = Not applicable

The program then displays the following prompts. The responses shown are in lowercase to indicate that they are specified by the user. If an error is made when entering a response, a correction can be made in one of the following three ways.

1. Press the BACKSPACE key or CTRL H keys to bring the cursor to the character in error,

2. Press the CTRL X keys to delete the entire line, or

3. Press the BREAK key to restart the program.

>DEVICE=dddd

Where:

dddd   is a 4-character string indicating one of the supported devices listed in the 8kb LSU device menu.   Configurations for these supported devices are automatically represented in the system and are listed in Table 2-7. If a unique device configuration that differs   from the   configurations   presented in Table 2-7 is desired, enter OTHR to the DEVICE= prompt. The type of device entered in the DEVICE= prompt determines which of the following four prompt sequences is issued:

●   Mag Tape Device

If an 8kb LSU-supported magnetic tape device is specified as a response, the following prompt is displayed on the system console:

>FILEMARKS=xxx

Where:

xxx   is a 1- to 3-digit decimal number from 0 through 255.   This number represents the number of filemarks to skip to reach the OS/32 file.   When the number of filemarks   is entered, followed by a carriage return, the 8kb LSU program loads OS/32 from the file | beginning after the last filemark skipped. |

- Hard Disk Device

    If an 8kb LSU supported disk device name is entered as a response, the following prompt is displayed on the system console:

    >VOL=vvvv,FILE=filename.ext

Where:

vvvv   is a 1- to 4-character volume name of the disk containing OS/32. The BOOTLOADER program reads and displays the existing volume name to the system console.

filename.ext

is a 1- to 8-character filename, followed by a 1- to 3-character extension. After filename.ext is entered, followed by a carriage return, the 8kb LSU program loads OS/32.

- Floppy Disk Device

    If FLPY is entered as a response, the following prompt is displayed on the system console:

    >DRV#=n

Where:

n   is a decimal number from 0 through 3 specifying the spindle containing the floppy disk from which the OS/32 is to be loaded. After the spindle number is specified, the following prompt is displayed:

    >VOL=vvvv,FILE=filename.ext

Where:

vvvv   is a 1- to 4-character volume name of the floppy disk containing OS/32. The BOOTLOADER program reads and displays the existing volume to the system console.

f ilename.ext

is a 1- to 8-character filename, followed by a 1- to 3-character extension. After filename.ext and a carriage return is entered, the BOOTLOADER program loads OS/32.

● Other

A response of OTHR to the DEVICE= prompt allows the user to specify a device address other than that listed in Table 2-7 for a particular device. The following prompt sequence is displayed on the system console:

>DEV ADDR=aaa

Where:

aaa   is a 1- to 3-digit hexadecimal number indicating the address of the device containing OS/32.

>DEV CODE=cc

Where:

cc   is a 2-digit hexadecimal number indicating the code of the device from which the operating system is to be loaded. The device codes listed in Table 2-7 are those used for 8kb LSU supported device names. The device code entered indicates the type of device that contains the operating system. The next prompt displayed in the prompt sequence is dependent upon the type of device indicated by the device code entry.

```
>CTLR ADDR=mmm
```

Where:

mmm   is a 1- to 3-digit hexadecimal number indicating the controller address for the device from which the operating system is to be loaded. If the device is a magnetic tape or a floppy disk, this prompt is not displayed.

```
>SLCH ADDR=sss
```

Where:

sss   is a 1- to 3-digit hexadecimal number indicating the SELCH address. If the device is a floppy disk, this prompt is not displayed.

```
>FILEMARKS=xxx
```

Where:

xxx   is a 1- to 3-digit decimal number from 0 through 255. This number represents the number of filemarks to skip to reach OS/32. This prompt is displayed only if the device code entered indicates a magnetic tape device.

```
>DRV#=n
```

Where:

n   is the decimal number specifying the spindle containing the floppy disk from which the OS/32 is to be loaded. This prompt is displayed only if the device code indicates a floppy disk device.

2.4.4.1 Examples of Loading OS/32 From Different Devices

● Loading OS/32 from an 8kb LSU-supported magnetic tape device:

```
BASIC TEST COMPLETE
3200 8KB LSU BOOTLOADER Rnn-uu
MG85
MGC5
MG62
DS5R
DS5F
DS67
D256
MM68
MM01
MM67
C13R
C13F
C40F
C67F
FLPY
OTHR
DEVICE=MG85
FILEMARKS=2
```

The above example, followed by a carriage return (CR), loads OS/32 from the third file on the tape.

● Loading OS/32 from an 8kb LSU-supported disk device:

```
BASIC TEST COMPLETE
3200 8KB LSU BOOTLOADER Rnn-uu
MG85
MGC5
MG62
DS5R
DS5F
DS67
D256
MM68
MM01
MM67
C13R
C134
C40F
C67F
FLPY
OTHR
DEVICE=DS5R
VOL=MT32,FILE=OS3220.KIR
```

The above example, followed by a CR, loads OS/32.

● Loading OS/32 from an 8kb LSU-supported floppy disk:

```
        BASIC TEST COMPLETE
        3200 8KB LSU BOOTLOADER Rnn-uu
        MG85
        MGC5
        MG62
        DS5R
        DS5F
        DS67
        D256
        MM68
        MM01
        MM67
        C13R
        C13F
        C40F
        C67F
        FLPY
        OTHR
        DEVICE=FLPY
        DRV#=1
        VOL=MT32,FILE=OS3220.KIR
```

The above example, followed by a CR, loads OS/32.

● Loading OS/32 from an 8kb LSU-supported disk device that has
  a nonstandard configuration:

```
        BASIC TEST COMPLETE
        3200 8KB LSU BOOTLOADER Rnn-uu
        MG85
        MGC5
        MG62
        DS5R
        DS5F
        DS67
        D256
        MM68
        MM01
        MM67
        C13R
        C13F
        C40F
        C67F
        FLPY
        OTHR
        DEVICE=OTHR
        DEV ADDR=EC
        DEV CODE=35
        CTLR ADDR=EB
        SLCH ADDR=FA
        VOL=MT32,FILE=OS3220.KIR
```

The previous example, followed by a CR, loads OS/32.

● Loading OS/32 from an 8kb LSU-supported magnetic tape device that has a nonstandard configuration:

```
BASIC TEST COMPLETE
3200 8KB LSU BOOTLOADER Rnn-uu
MG85
MGC5
MG62
DS5R
DS5F
DS67
D256
MM68
MM01
MM67
C13R
C13F
C40F
C67F
FLPY
OTHR
DEVICE=OTHR
DEV ADDR=85
DEV CODE=40
SLCH ADDR=F0
FILEMARKS=2
```

The above example, followed by a CR, loads OS/32 from the |
third file on the tape. |

- Loading OS/32 from an 8kb LSU-supported floppy disk device that has a nonstandard configuration:

```
BASIC TEST COMPLETE
3200 8KB LSU BOOTLOADER Rnn-uu
MG85
MGC5
MG62
DS5R
DS5F
DS67
D256
MM68
MM01
MM67
C13R
C13F
C40F
C67F
FLPY
OTHR
DEVICE=OTHR
DEVICE ADDR=C2
DEVICE CODE=37
DRV#=1
VOL=MT32,FILE=OS3220.KIR
```

The above example, followed by a CR, loads OS/32.

2.4.4.2   8kb Loader Storage Unit (LSU) Messages

The following messages can be generated if errors occur using the 8kb LSU:

```
BASIC CONFIDENCE TEST ERRORS

CHECKSUM ERROR

CANNOT CLEAR PSW

EPSR ARGUMENT 00000001, L FLAG NOT SET

EPSR ARGUMENT 00000001, ALSO SET C, V, OR G

EPSR DOESN'T PROPERLY UNLOAD PSW

LA---RX1 FAILURE

LA---RX2 FAILURE
```

THI---CC NOT CLEAR

THI---1ST OP REGISTER CHANGED

THI---CC NOT SET

OR---INCORRECT RESULT

NR---INCORRECT RESULT

SR---INCORRECT RESULT

EXBR---INCORRECT RESULT

SRLS---INCORRECT RESULT

MHR---INCORRECT RESULT

DH---INCORRECT RESULT

> If any of the above BASIC CONFIDENCE TEST error messages appear, except CHECKSUM ERROR, the user should contact his service representative.

IOERROR DU

> The user-specified device is unavailable.

IOERROR UNRE

> The I/O error that occurred is unrecoverable.

IOERROR CONFG

> An I/O error occurred because the hardware configuration does not correspond with that specified by the user.

FILE NOT FOUND

> User-specified fd does not exist, or file is not contiguous.

MEMTST ERR nnnnnn

> An error occurred during a memory test before OS/32 was loaded. The faulting memory location is at nnnnnn.

DISC NOT INITIALIZED

> Disk pack (fixed or removable) is not properly initialized.

2.5 MEMORY CONFIGURATION ERROR

When a memory configuration error is detected at system startup time, the following message is displayed on the system console:

        MEMORY CONFIGURATION ERROR xx BLOCK yy
        MEMORY ERROR RECORDING DISABLED

Where:

    xx   contains one of the following codes:

            • 01 indicates error occurred during initial
               clearing of error logger status bits.

            • 02 indicates error occurred during
               verification of controller address
               range.

            • 03 indicates error occurred during
               verification of memory interleaving.

            • UN indicates memory error recording was
               specified at sysgen but the MCONFIG.OBJ
               file was not included in sysgen. Occurs
               only on systems generated using the
               configuration utility program (CUP) R06.

    yy   specifies the memory block where the error was
                    detected.

Errors 01, 02, or 03 can result from improper specification of physical memory at sysgen time or a hardware malfunction. If an error occurs, check the memory configuration statements used at sysgen; if no incorrect statements are present, check the hardware using the appropriate hardware diagnostic.

NOTE

    Disabling the memory error recording does
    not interfere with the operating system's
    ability to function. However, memory
    errors will not be recorded.

2.6 MODIFYING OS/32 FOR OTHER PERIPHERAL CONFIGURATIONS

Both starter systems are configured with standard device numbers.
If your particular configuration is not standard, these
procedures can be used to change device addresses after the
appropriate starter system is loaded.

2.6.1 Modification of the Console Device Address

To modify the address of the system console device to that of a
device of similar attributes (for instance, from TTY @ DN=02 to
TTY @ DN=22 or from TTY @ DN=10 to video display unit (VDU) @
DN=12), follow this procedure:

1. Obtain the address of the device mnemonic table (DMT) from
    the library loader map of starter system (entry point DMT).

2. Add X'C' to this address.

3. Read from memory the fullword contents specified by this
    address to obtain the device control block (DCB) address.

4. Add X'1A' to the DCB address.

5. Modify the halfword at this address to contain the new
    physical address of the console device.

6. Press DTA-60-ADD-RUN. The operating system id should be
    displayed at the system console.

For Perkin-Elmer Series 3200 processors, enter:

    @60
    >

To modify the system console from one type of device to another
(for instance, from TTY @ DN=02 to VDU @ DN=10), follow this
procedure:

1. Obtain the address of the DMT from the library loader map of
    the starter system (entry point DMT).

2. Find the address of the current system console by reading
    fullword at A(DMT) + X'C'.

3. Scan down the DMT starting at A(DMT) + X'10', every 8 bytes
    for the device mnemonic with the device characteristics
    desired. For example, CRT would be fullword 43525420. |

4. When found, swap the DCB address in the following fullword with that of the system console.

5. If the device number is not correct, use the preceding procedure to modify it.

6. Restart operating system at location X'60'.

2.6.2 Modification of Nonconsole Device Addresses

To modify the address of a nonconsole device, follow this procedure:

1. Obtain the DCB address from the library loader map of starter system. Each DCB in the system has an address specified on the map.

   Format:

       DCBdddnn

   Parameters:

ddd	is the device code.
nn	is an index indicating which DCB it represents.

   Example:

DCB01600	is the DCB associated with the teletype device (device code = 16). It is the first DCB generated for this teletype device, hence an index of X'00'.

2. Add X'1A' to the DCB address.

3. Modify the halfword at this address to contain the physical address of the device. The system console should be used to perform this modification. For example:

       BIAS   0
       MODIFY   1234,0034

   See Chapter 3 for an explanation of the BIAS and MODIFY commands.

2.6.3 Modification of the SELCH Address

To modify the address of a SELCH, follow this procedure:

1. Obtain the associated DCB address as in Section 2.5.1.

2. Add X'98' to this address.

3. Modify the halfword at the resulting location to contain the correct SELCH physical address. The system console should be used to perform this modification.

2.6.4 Modification of the Controller Address

To modify the address of a controller, follow this procedure:

1. Obtain the associated DCB address as stated in Section 2.5.2.

2. Add X'9A' to this address.

3. Modify the halfword at the resulting location to contain the controller physical address. Use the system console to perform this modification.

2.6.5 Modification of the Console Device Name/Type

If the console is a VDU device, rather than a TTY device or vice versa, follow this procedure:

1. Obtain the address of the initial value table from the library loader map of the starter system (entry point IVTBL).

2. Read the two halfwords from memory at this location. The contents at this location should be:

    434F,4E20 (CON in 7-bit ASCII)

3. Modify these four bytes to contain the desired console device name:

    4352,5420 (CRT in 7-bit ASCII) |

4. Press DTA-60-ADD-RUN. The operating system id and license or
    contract number should be displayed on the VDU.   For
    Perkin-Elmer Series 3200 processors, enter:

        @ 60
        >

                              NOTE

            This procedure should be followed only
            immediately after system initialization.

To avoid repeating this procedure for each system initialization,
the operating system should be patched by OS/32 Patch.

This procedure assumes the device address is correct. If it is
not, perform the procedure to change the address.

CHAPTER 3
CONSOLE USE AND OPERATOR COMMANDS

3.1   INTRODUCTION

Communication between the operator and the operating system or
user program is accomplished by an operator command language.
The operator commands described in this chapter are input to the
system via the system console.

3.2 ALLOCATE COMMAND

The ALLOCATE command is used to create a direct access file or a data communications line control block (LCB) for a buffered terminal manager.

Format:

$$
\text{ALLOCATE fd,}
\begin{cases}
\text{CONTIGUOUS,fsize}\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right] \\[2ex]
\text{EC}\left[\left[/\begin{Bmatrix}\text{bsize}\\64\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{isize}\\3\end{Bmatrix}\right]\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right]\right] \\[2ex]
\text{INDEX}\left[,\begin{Bmatrix}\text{lrecl}\\126\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{bsize}\\1\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{isize}\\1\end{Bmatrix}\right]\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right] \\[2ex]
\text{NB}\left[,\begin{Bmatrix}\text{lrecl}\\126\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{bsize}\\64\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{isize}\\3\end{Bmatrix}\right]\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right] \\[2ex]
\text{ITAM}\left[,\begin{Bmatrix}\text{lrecl}\\80\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{bsize}\\1\end{Bmatrix}\right]\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right]
\end{cases}
$$

Parameters:

fd	is the file descriptor of the device or file to be allocated.
CONTIGUOUS	specifies that the file type to be allocated is contiguous.
fsize	is a decimal number indicating file size, which is required for contiguous files. It specifies the total allocation size in 256-byte sectors. This size may be any value up to the number of contiguous free sectors existing on the specified volume at the time the command is entered.

keys   specifies the write and read protection keys for the file. These keys are in the form of a hexadecimal halfword, the most-significant byte of which signifies the write key and the least-significant byte, the read key. If this parameter is omitted, both keys default to 0.

EC   specifies that the file type to be allocated is extendable contiguous.

bsize   is a decimal number specifying the physical block size to be used for buffering and debuffering operations on the index file or data communications device, and for the data blocks used for indexed, nonbuffered indexed, and extendable contiguous files. When INDEX, EC, or NB is specified, bsize represents the block size in sectors of the physical data blocks containing the file. When ITAM is specified, bsize represents the buffer size in bytes. For INDEX files and ITAM buffers, this parameter cannot exceed the maximum block size established by the system generation (sysgen) procedure. For EC and NB files, this parameter may be any value between 1 and 255 inclusive. If bsize is omitted, the default value for INDEX files and ITAM buffers is 256 bytes (one sector). For EC and NB files, the default is 64 sectors.

isize   is a decimal number specifying the index block size. For INDEX files, the default value is one sector (256 bytes). For EC and NB files, the default value is three sectors (768 bytes). The index block size cannot exceed the maximum disk block size established by the sysgen procedure. Neither bsize nor isize may exceed 255.

INDEX   specifies the file type to be allocated is indexed.

lrecl   is a decimal number specifying the logical record length of an indexed or nonbuffered indexed file, or a data communications device. It cannot exceed 65,535 bytes. The default for indexed and nonbuffered indexed files is 126. It may optionally be followed by a slash (/), which delimits lrecl from bsize.

NB   specifies that the file type to be allocated is nonbuffered indexed.

ITAM   specifies that the device to be allocated is a data communications device.

Functional Details:

To assign an indexed file, sufficient room must exist in system
space for two data block buffers plus one index block buffer,
each of the stated size. Therefore, if bsize is very large, the
file may not be assignable in some memory-bound situations. At
sysgen time, a maximum block size parameter is established in the
system, and bsize cannot exceed this constant.

To assign an EC or NB file, sufficient room must exist in system
space to contain only the indexed block of the stated size. The
data blocks for EC and NB files are not buffered in system space,
and, thus, are not constrained to the sysgened block size.

A file associated with a nonzero account number cannot be
allocated from the system console.

Examples:

AL JANE.TSK,CO,64	allocates on the system volume a contiguous file named JANE.TSK which has a total length of 64 sectors (16kb) with protection keys of 0.
AL VOL1:AJM.OBJ,IN,126	allocates on volume VOL1: an indexed file named AJM.OBJ which has a logical record length of 126 bytes. The buffer size and index block size default to 1 sector and the protection keys default to 0.
AL M300:AJM.BLK,IN,132/4	allocates on volume M300: an indexed file named AJM.BLK which has a logical record length of 132 bytes, data block size of 4 sectors, and default index block size (isize) of 1 sector. The protection keys default to 0. When this file is assigned, the system must have 2.5kb of available system space for buffers.
AL THISFILE,IN,256/4/2	allocates on the system volume an indexed file named THISFILE (blank extension) which has a logical record length of 256 bytes, a data block size of 4 sectors, an index block size of 2 sectors, and protection keys of 0.

AL VO1:AJM.OBJ,IN,126//3   allocates on volume VO1: an indexed file named AJM.OBJ which has a logical record length of 126 bytes. The data block size defaults to 1 sector, the index block size is 3 sectors, and the protection keys default to 0.

AL SYS:XFILE.DTA,EC   allocates on volume SYS: an extendable contiguous file named XFILE.DTA with default data block size of 64 sectors and index block size of 3 sectors. The file initially contains no records, and has a record length of one sector (same as a contiguous file).

AL YFILE.DAT,NB,240/250/5   allocates on the system volume a nonbuffered indexed file named YFILE.DAT with logical record length of 240 bytes, data block size of 250 sectors, and index block size of 5 sectors. The file initially contains no records.

Messages:

ALLO-ERR TYPE=

    indicates that allocation failed for reasons denoted by TYPE field (i.e., displayed to the right of the equal sign). See Appendix B for possible entries in the TYPE field.

FD-ERR

    indicates an invalid file descriptor (fd).

NODA-ERR

    indicates that direct access support was not included in the system at sysgen.

NOPR-ERR

    indicates that a required operand is missing.

PARM-ERR

    indicates an operand syntax error.

```
-------------
|           |
|   APU     |
|           |
-------------
```

3.3 APU CONTROL AND DISPLAY COMMAND

The APU command is used to control the state of an auxiliary processing unit (APU) within a 3200MPS System.

This command is valid only for Model 3200MPS Systems.

Format:

```
              / LPU                     \
             |                           |
             |    [ /START    \ ]        |
             |    | |STOP      | |        |
             |    | |ON        | |        |
             |    | |OFF       | |        |
  APU    <    n | ,<ENABLE     > |   >
             |    | |DISABLE   | |        |
             |    | |LPU=n/ADD | |        |
             |    | |LPU=n/DEL | |        |
             |    [ \LPU=*/DEL/ ]        |
              \                         /
```
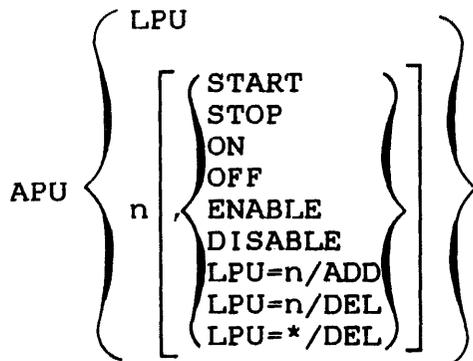
Parameters:

LPU   causes the LPU-to-APU mapping table to be displayed.

n   is the APU number to which the command is directed for control or display. If no subsequent parameter is included, the status of the APU is displayed.

START   starts the APU.

STOP   forces the APU idle.

ON   marks the APU available for logical processing unit (LPU) assignment.

OFF   stops the APU, forces all tasks from that APU to the central processing unit (CPU) for execution, and marks the APU off.

ENABLE   from the disabled state, the enable parameter
                        performs a power-up link check with the APU
                        and marks the APU to the OFF state. If the
                        APU is not in the DISABLED state, this
                        parameter is rejected and the error message,
                        SEQ-ERR, is displayed.

DISABLE   marks the APU to the DISABLED state if it is
                        currently in the OFF state. If the APU is not
                        currently in the OFF state, the parameter is
                        rejected and a SEQ-ERR error message is
                        displayed. The disabled state is used when
                        powering down an APU and performing
                        maintenance.

LPU=n/ADD   causes the LPU specified by n to be mapped to
                        the APU specified by the preceding APU n
                        parameter.

LPU=n/DEL   deletes from the mapping table the logical
                        association between the LPU, specified by n,
                        and the APU specified by the preceding APU n
                        parameter. The LPU is remapped to the CPU.

LPU=*/DEL   deletes from the mapping table all LPUs
                        currently associated with the APU specified by
                        APU n. These LPUs are then remapped to the
                        CPU.

Functional Details:

This command gives the operator control over the APUs of the
system. Entering APU LPU will display the range of LPU numbers
(0-N), the range of APU numbers (1-M), and LPU/APU mapping. LPUs
mapped to the CPU (processor 0) are omitted from the display.

Example:

    *APU LPU
        LPU: 0-24 APU: 1-4
        LPU/APU: 2/2 3/1 4/1 19/3 21/4

In the above example, 25 LPUs (0-24) exist in a system consisting
of a CPU and four APUs (1-4). LPUs 2, 3, 4, 19, and 21 are
mapped to APUs 2, 1, 1, 3, and 4, respectively. All other LPUs
are mapped to the CPU.

Entering the appropriate APU number will display the following
items:

- Mapping status

  ON, available for assignment of LPUs.

  OFF, not available for assignment of LPUs

- Number of LPUs assigned

- Execution status

ACTIVE	running a task (called current task)
ERROR-IDLE	idle because of a hardware error while running the current task
Q-WAIT	idle because of absence of tasks available for running (ready queue empty)
EXCLUSIVE=	idle because of expecting a task to return from the CPU

- Names of associated tasks

CURRENT=taskname	the current task name. This task is currently being executed on the APU.
EXCLUSIVE=taskname	the name of the task with exclusive APU use. No other task is allowed to execute on the APU until this task has completed.
MAPPING=taskname	the name of the task with mapping rights for the APU.
CONTROL=taskname	the name of the task with control rights for the APU.
WAIT=taskname	the name of the task expected to return from the CPU before resuming execution.
READYQ=taskname,list	names of tasks in the ready queue

Examples:

```
*APU 1
   ON, ASSIGNED 3 LPU, ACTIVE
   CURRENT=EXEC1
   READYQ=FORT3, APPLIQ3, APPLIQ2, SOAKER
```

APU 1 is marked on and is assigned three LPUs. A display of
these assigned LPUs can be obtained by entering APU LPU (see the
preceding example). Task (EXEC1) is currently executing and the
ready queue on the APU includes tasks FORT3, APPLIQ3, APPLIQ2,
and SOAKER.

```
*APU 2
    ON, ASSIGNED 2 LPU, ERROR_IDLE
    CURRENT=WCSSOP, MAPPING=EXEC, CONTROL=EXEC
    READYQ=APPLIQ1
```

APU 2 is marked on and is assigned two LPUs. Task (WCSSOP) is
suspended because of a hardware error (CPU reaction is expected).
A task (EXEC) possesses both mapping and control rights on the
APU and the ready queue of the APU includes only task APPLIQ1.

```
*APU 3
    OFF
```

APU 3 is marked off.

```
*APU 4
    ON, ASSIGNED 1 LPU, Q_WAIT
```

APU 4 is on and assigned one LPU. It is waiting for a task to be
transferred to it.

```
*APU 5
    ON, ASSIGNED 4 LPU, TASK_WAIT
    WAIT=EXEC1
    READYQ=TASK35
```

APU 5 is on and assigned four LPUs. It is waiting for the task
EXEC1 to be returned to it to resume execution.

```
*APU 6
    ON, ASSIGNED 1 LPU, ACTIVE
    EXCLUSIVE=WCSSOP
```

APU 6 is on and assigned one LPU. It is active and the APU is
assigned exclusively to the task WCSSOP.

```
*APU 6, LPU=8/ADD
```

LPU 8 is mapped to APU 6.

48-030 F00 R01   3-9

| *APU 7, LPU=8/DEL

| The logical association between APU 7 and LPU 8 is deleted from
| the mapping table. LPU 8 is remapped to the CPU.

| *APU 5, */DEL

| The logical association between all LPUs associated with APU 5
| are deleted from the mapping table. The LPUs are remapped to the
| CPU.

| To use the ENABLE parameter, the APU must be waiting for a
| power-up link check as a result of local clear (SCLRO) or
| power-up on the APU. The APU fault/power lamp is red. If the
| ENABLE command is successful, the APU fault/power lamp is green.

| **Messages:**

| APU-ERR

| indicates the command processor was unable to gain access to
| the APU.

| FORM-ERR

| indicates a command syntax error.

| MAP-ERR

| indicates the LPU is already mapped to an APU.

| PARM-ERR

| indicates an operand syntax error.

| SEQ-ERR

| indicates that required control function is invalid: a
| start/stop is required for a marked-off APU.

3.4 ASSIGN COMMAND

The ASSIGN command assigns a device, file, or data communications
device to one of a task's logical units.

Format:

$$
\text{ASSIGN lu,fd} \left[, \left[\left\{ \begin{array}{c} \text{(access privileges)} \\ \text{SRW} \\ \text{SREW} \\ \text{SRO} \end{array} \right\} \right] \quad \left[\left\{ \begin{array}{c} \text{keys} \\ \text{0000} \end{array} \right\} \right] \quad \left[, \left\{ \begin{array}{c} \text{SVC15} \\ \text{SVCF} \\ \text{VFC} \\ \text{HI} \\ \text{LOW} \\ \text{MEDIUM} \end{array} \right\} \right] \right]
$$
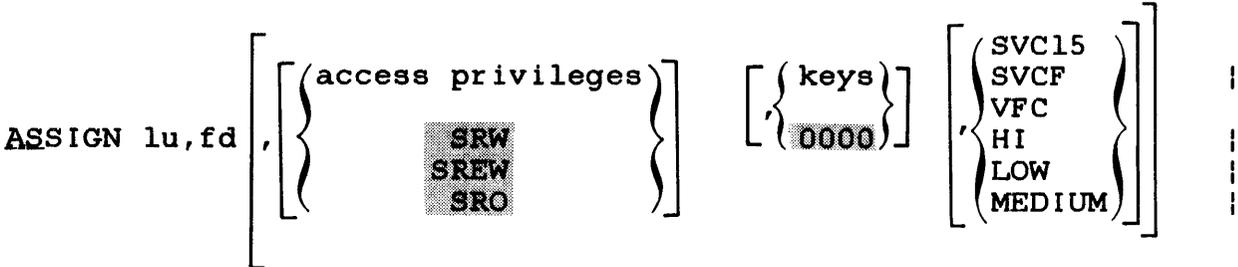
NOTE

If the access privileges and keys
parameters are omitted and SVC15, VFC,
HI, LOW, or MEDIUM are specified, the
respective positional commas belonging to
the omitted parameters can also be
omitted if desired.

If the access privileges and SVC15, VFC,
HI, LOW, or MEDIUM parameters are
specified and the keys parameter is
omitted, the positional comma belonging
to the keys parameter can be omitted if
desired.

Parameters:

lu   is a decimal number specifying the logical
                     unit number to which a device or file is to be
                     assigned.

fd   is the file descriptor of the device or file
                     to be assigned.

access privileges	specifies the desired access privileges. The default access privileges are:

- SRW for contiguous files,

- SREW for index, EC, and NB files,

- SRW for devices, and

- SRO for files with a nonzero account number.

keys	signifies the write/read protection keys of the file or device to be assigned.
SVC15 SVCF	signifies that the specified device is to be assigned for SVC15 access. This option pertains to data communications devices only. If SVC15 access is specified, vertical forms control (VFC) cannot be specified.
VFC	specifies the use of vertical forms control for the assigned logical unit (lu). If this parameter is omitted, there is no vertical forms control for the device assigned to the specified lu.
HI	indicates that the assigned magnetic tape will operate at the GCR density rate of 6250 bpi. This parameter is applicable for 6250 tape drives only.
LOW	indicates that the assigned magnetic tape will operate at the NRZI density rate of 800 bpi. This parameter is applicable for 6250 tape drives only.
MEDIUM	indicates that the assigned magnetic tape will operate at the phase encoded (P-E) density rate of 1600 bpi. This parameter is applicable for 6250 tape drives only.

Functional Details:

Access privileges can be one of the following:

SRO   Sharable read-only

ERO   Exclusive read-only

SWO   Sharable write-only

EWO   Exclusive write-only

SRW   Sharable read/write

SREW   Sharable read, exclusive write

ERSW   Exclusive read, sharable write

ERW   Exclusive read/write

The command is rejected if the requested access privilege cannot be granted.

A file with a nonzero account number can only be assigned with access privileges of SRO or ERO. If the access privileges field is omitted, a default of SRO is assumed for a file with a nonzero account number, and SRW or SREW is assumed for system files. The DISPLAY LU command can be used to determine the current access privileges of all assigned units. See Section 3.21 for more information.

When a file has been assigned to a task, the operator might want to prevent other tasks from accessing that file while it is being used. For this reason, the user can ask for exclusive access privileges, either for read or write, at assignment time. This protection is called dynamic because it is only in effect while the file remains assigned.

A file cannot be assigned with a requested access privilege if that privilege is incompatible with some other existing assignment to that file. For example, a request to open a file for EWO is compatible with an existing assignment for SRO or ERO but is incompatible with any existing assignment for other access privileges. Table 3-1 illustrates compatibilities and incompatibilities between access privileges.

TABLE 3-1   ACCESS PRIVILEGE COMPATIBILITY

	ERSW	ERO	SRO	SRW	SWO	EWO	SREW	ERW
ERSW	-	-	-	-	*	-	-	-
ERO	-	-	-	-	*	*	-	-
SRO	-	-	*	*	*	*	*	-
SRW	-	-	*	*	*	-	-	-
SWO	*	*	*	*	*	-	-	-
EWO	-	*	*	-	-	-	-	-
SREW	-	-	*	-	-	-	-	-
ERW	-	-	-	-	-	-	-	-

*   Compatible
-   Incompatible

The keys field is in the format of a 4-digit hexadecimal number. The left two digits signify the write protection key, and the right two digits, the read protection key. If omitted, the default is 0000. These keys are checked against the appropriate existing keys for the file or device. The command is rejected if the keys are invalid. The keys associated with a file are specified at file allocation time. They can be changed by a REPROTECT command or through a supervisor call (SVC) 7 reprotect function call.

If the values of the keys are within the range of X'01' to X'FE', the file or device cannot be assigned for read or write access unless the operator or requesting task supplies the matching keys. If a key has a value of X'00', the file or device is unprotected for that access mode. Any key supplied is accepted as valid. If a key has a value of X'FF', the file is unconditionally protected for that access mode. It cannot be assigned for that access mode to any user task (u-task) or diagnostic task (d-task), regardless of the key supplied. An unconditionally protected file may be assigned to an executive task (e-task), including the system manager.

A file can be deleted only if its write and read protection keys are zero (X'0000'). If the keys are X'FF00', for example, the file cannot be deleted.

Some examples of protection keys are shown in Table 3-2.

TABLE 3-2   PROTECTION KEYS AND THEIR MEANINGS

WRITE KEY	READ KEY	MEANING
00	00	Completely unprotected
FF	FF	Unconditionally protected
07	00	Unprotected for read, conditionally protected for write (user must supply write key=X'07')
FF	A7	Unconditionally protected for write, conditionally protected for read
00	FF	Unprotected for write, unconditionally protected for read
27	32	Conditionally protected for both read and write

The protection keys of a device can be changed by the system operator by the REPROTECT command.

At the time of assignment, an assigned direct access file is positioned at the end of the file for access privileges SWO and EWO. It is positioned at the beginning of the file for all other access privileges. The ASSIGN command is rejected if the specified lu is assigned and the currently selected task is not dormant. To reassign an lu for an active task, the lu must first be closed.

Examples:

AS 3,CR:,ERO   assigns the card reader to lu 3 with ERO access privilege. The keys default to 0000.

AS 2,FILE.DAT,EWO   assigns a disk file to lu 2. The EWO access privilege causes the file to be positioned at the end. New records are appended.

AS 2,TEST.JOB,VFC   assigns a disk file to lu 2. VFC is in use. Access privileges and keys parameters are omitted along with their respective commas.

```
AS 2,TEST.JOB,,,VFC          assigns a disk file to lu 2. VFC is
                             in use.  Access privileges and keys
                             parameters  are  omitted,  but
                             positional commas are specified.

AS 2,TEST.JOB,,OOFF,VFC      assigns a disk file to lu 2.  VFC is
                             in use.  The  positional  comma
                             belonging  to the omitted access
                             privileges  parameter  must  be
                             specified.

AS 2,TEST.JOB,SRO,VFC        assigns a disk file to lu 2.  VFC is
                             in use. The keys  parameter,  along
                             with  the  positional  comma,  is
                             omitted.

AS 2,TEST.JOB,OOFF,VFC       is an invalid assignment because the
                             positional comma belonging to  the
                             omitted access privileges parameter
                             must be specified.

AS 2,TEST.JOB,SRO,VFC,SVC15

                             is an invalid assignment because VFC
                             and  SVC 15 access are  mutually
                             exclusive and cannot be specified in
                             the same assignment.

AS 2,MAG1:,LOW               assigns a mag tape drive  to  lu  2.
                             The low parameter indicates that the
                             drive  will  operate  at  the  NRZI
                             density rate of 800 bpi.

AS 2,MAG1:,SRW,MEDIUM        assigns a mag tape drive  to  lu  2.
                             The MEDIUM parameter indicates that
                             the drive will operate  at  the  P-E
                             density rate of 1600 bpi.

AS 2,MAG1:,,,HI              assigns a mag tape drive  to  lu  2.
                             The HI parameter indicates that the
                             drive  will  operate  at  the  GCR
                             density  rate  of 6250 bpi.  Access
                             privileges and keys  parameters  are
                             omitted,  but  positional commas are
                             specified.

AS 2,MAG1:,SRW,LOW,SVCF      is  an  invalid  assignment  because
                             density  select  and SVCF access are
                             mutually  exclusive  and  cannot  be
                             specified in the same assignment.
```

Messages:

ASGN-ERR TYPE=

    indicates the assign failed for reason noted by the TYPE field. See Appendix B for possible entries in the TYPE field.

FD-ERR

    indicates an invalid fd.

LU-ERR

    indicates that an invalid lu number or lu was assigned.

NODA-ERR

    indicates there is no direct access support in the system.

NOPR-ERR

    indicates an invalid operand.

PARM-ERR

    indicates an operand syntax error.

PRIV-ERR

    indicates an invalid access privilege mnemonic.

SEQ-ERR

    indicates that task was not paused or is dormant.

SPAC-ERR

    indicates the task exceeds established maximum system space usage.

TASK-ERR

    indicates that there is no currently selected task.

```
 ------------
|    ATTN    |
 ------------
```

3.5 ATTN COMMAND

The ATTN command is used to modify the priority of the system task that processes operator and command substitution system (CSS) commands.

Format:

$$\text{ATTN} \left[\left\{ \begin{array}{c} n \\ 2 \end{array} \right\} \right]$$

Parameter:

n   is a decimal number from 2 through 249
                    specifying the new priority of the command
                    processor task. If n is omitted, the default
                    is 2.

Functional Details:

The ATTN command should be entered only when critical real-time tasks are executing in the foreground. It lowers the priority of the task which processes system commands, and therefore, could delay execution of a time-sensitive operator command. This command can be entered only from the system console.

Example:

ATTN 120   sets the priority of the system command
                    processor to 120.

Message:

ATTN-ERR

    indicates that n is not a decimal number from 2 through 249.

3.6 BFILE COMMAND

The BFILE command backspaces to the preceding filemark on magnetic tapes, cassettes, and direct access files.

Formats:

BFILE fd   Used for magnetic tapes and cassettes only

BFILE fd [,lu]   Used for disk devices only

Parameters:

fd   is the file descriptor of the device or file to be backspaced to a filemark.

lu   is the logical unit to which the file is assigned.

Functional Details:

For magnetic tapes and cassettes, only the parameter fd should be specified; for direct access files, lu optionally can be specified. The account number must be 0 if specified.

Before entering the format for a file, the task using that file must be selected as the current task through the TASK command.

Examples:

BF MAG2:   causes the device MAG2: to backspace one filemark.

BF M300:AJM.OBJ,4   causes file AJM.OBJ that is assigned to lu 4 on volume M300 to backspace one filemark.

Messages:

ASGN-ERR TYPE=

      indicates that the file or device could not be assigned for the reason noted in the TYPE field. See Appendix B for possible entries in the TYPE field.

FD-ERR

      indicates that the specified fd has a syntax error or a nonzero account number.

IO-ERR

      indicates that an input/output (I/O) error or an illegal or unassigned lu was encountered.

LU-ERR

      indicates that lu was not a valid decimal number or was greater than maximum lu (maxlu) for the task.

NOBC-ERR

      indicates that bulk file command support is not included in the operating system.

NOPR-ERR

      indicates that no operand was specified.

TASK-ERR

      indicates that there was no currently selected task, and a BFILE command with an lu specified was entered.

3.7 BIAS COMMAND

The BIAS command is used to set a bias address for the EXAMINE and MODIFY commands.

Format:

$$\text{BIAS} \begin{Bmatrix} \text{address} \\ * \\ \text{*taskid} \end{Bmatrix}$$

Parameters:

address   is a hexadecimal bias to be added to the address given in any subsequent EXAMINE or MODIFY command. If address is omitted, all addresses specified in subsequent EXAMINE and MODIFY commands are treated as unbiased; they are assumed to be absolute physical addresses.

*   specifies that the bias is set to be the physical address of the first location of the currently selected task. If the currently selected task is rollable, specification of * results in an error.

*taskid   specifies that the bias is set to be the physical address of the first location of taskid. If this task is rollable, specification of BIAS *taskid results in an error. The currently selected task is not changed.

Functional Details:

A BIAS command overrides all previous BIAS commands. The operator should enter a BIAS command if the current value is unknown.

Examples:

TA EXAM   sets the currently selected task to task EXAM.

BI *   sets the bias to first location of task EXAM.

BI *EXAM   sets the bias to first location of task EXAM.

Messages:

PARM-ERR

indicates an operand syntax error.

ROLL-ERR

indicates BIAS * was entered and the currently selected task is rollable, or BIAS *taskid was entered and taskid is a rollable task.

TASK-ERR

indicates BIAS * was specified but there is no currently selected task, or BIAS *taskid was entered and no task with name taskid exists.

3.8   BRECORD COMMAND

The BRECORD command backspaces to the preceding record on magnetic tapes, cassettes, and direct access files.

Formats:

BRECORD fd	Used for magnetic tapes and cassettes only
BRECORD fd [,lu]	Used for disk files only

Parameters:

fd	is the file descriptor of the device to be backspaced one record.
lu	is the logical unit to which the file is assigned.

Functional Details:

For magnetic tapes and cassettes, only the parameter fd should be specified; for direct access files, lu optionally can be specified. The account number must be 0 if specified.

Before entering the format for disk devices, the task using the file must be selected as the current task through the TASK command.

Examples:

BR MAG2:	causes the device MAG2: to backspace one record.
BR M300:AJM.OBJ,4	causes the file AJM.OBJ that is assigned to lu 4 on volume M300 to backspace one record.

Messages:

ASGN-ERR TYPE=

indicates that the file or device could not be assigned for the reason noted in the TYPE field. See Appendix B for possible entries in the TYPE field.

FD-ERR

indicates that the specified fd has a syntax error or a nonzero account number.

IO-ERR

indicates that an I/O error or an illegal or unassigned lu was encountered.

LU-ERR

indicates that lu was not a valid decimal number or was greater than maxlu for the task.

NOBC-ERR

indicates that bulk file command support is not included in the operating system.

NOPR-ERR

indicates that no operand was specified.

TASK-ERR

indicates that there was no currently selected task when a BRECORD command with an lu specified was entered.

3.9 BUILD AND ENDB COMMANDS

The BUILD and ENDB commands copy data from the command stream to a device or file (account number must be zero if specified). These commands are normally used from a CSS file but can also be entered from the system console.

Format:

        BUILD fd [,APPEND]
           .
           .
           .
        ENDB

Parameters:

        fd   is the file descriptor of the device or file
                            to which data is to be copied. If fd does not
                            contain an extension, .CSS is used as a
                            default. If a blank extension is desired, the
                            period following the filename must be typed.
                            If fd refers to a direct access file, an index
                            file by that name is allocated with a logical
                            record length equal to the sysgened command
                            buffer length, a blocksize of 1, and keys of
                            0000. If the specified fd already exists,
                            that fd is deleted and a new fd is allocated.

        APPEND   allows the user to append data to an existing
                            fd. If the fd does not exist, it is
                            allocated.

Functional Details:

Lines entered from the console following BUILD are not treated as commands, but as data, and are copied to the specified device or file until an ENDB is encountered. The BUILD command must be the last command on an input line. Further data appearing on that line is treated as a comment and causes no action to be taken. The BUILD command can be entered from the console only if no CSS files are active. It can also be entered from a CSS file. If a BUILD command is entered at the console, the prompt .CMDP> indicates that the system command processor is requesting data.

The ENDB command must appear as the first command on a line and may be preceded only by spaces. Any commands following the ENDB on the same command line are executed. No error response is possible from ENDB. If ENDB is not entered as the first command on the command line, the line is copied to the BUILD file.

Messages:

ASGN-ERR

    indicates that an output file or device could not be assigned for the reason noted in the TYPE field. See Appendix B for possible entries in the TYPE field.

FD-ERR

    indicates invalid fd or no index file support.

IO-ERR

    indicates that an I/O error was encountered on an output file or device.

SEQ-ERR

    indicates that CSS file is active, BUILD entered from system console.

3.10 CANCEL COMMAND

The CANCEL command terminates the task as if the task had executed an SVC 3 and terminated with an end of task code of 255.

Format:

    CANCEL taskid

Parameter:

    taskid   is required if the command is entered from the
                    system console, but is optional if used in
                    CSS.

Functional Details:

If the task is nonresident, it is removed from the system. All outstanding I/O is terminated and the task's logical units are closed. If the task is resident, it is not removed from memory. Its logical units are not closed, but are checkpointed. This command can be entered even when the currently selected task is dormant. It has no effect on a resident task that has already gone to end of task, unless preceded by an OPTIONS NONRESIDENT command. If preceded by an OPTIONS NONRESIDENT command, it will remove a task from memory. The normal response to this command is:

    hh:mm:ss   TASKID:END OF TASK 255

For users of the Model 3200MPS System, the following information |
applies: |

The CANCEL command can be entered when the currently selected |
task belongs to either the CPU or an APU. If an APU has been |
waiting for a cancelled task, it is restarted using the next task |
in its ready queue. |

Messages:

PARM-ERR

    indicates that taskid is missing or task is not loaded.

| SEQ-ERR

| indicates that task was not dormant or paused.

TASK-ERR

    indicates that there is no currently selected task. This applies to CSS only.

3.11 CLOSE COMMAND

The CLOSE command permits the operator to close (unassign) one or more files or devices assigned to the currently selected task's logical units.

Format:

$$\text{CLOSE} \quad \left\{ \begin{array}{c} lu_1 \left[, lu_2, \ldots, lu_n \right] \\ ALL \end{array} \right\}$$

Parameters:

lu   is a decimal number specifying the lu assignments.

ALL   specifies that all logical units of the currently selected task are to be closed.

Functional Details:

Closing an unassigned lu does not produce an error message. A CLOSE command can be entered only if the referenced task is dormant or paused.

Examples:

CL 1,3,5   closes logical units 1, 3, and 5 of the currently selected task.

CLOSE A   closes all logical units of the currently selected task.

Messages:

CLOS-ERR

    indicates that close failed for reason denoted by TYPE field.

PARM-ERR

    indicates an invalid or missing parameter.

SEQ-ERR

    indicates that task was not dormant or paused.

TASK-ERR

    indicates that there is no currently selected task.

3.12   CONTINUE COMMAND

The CONTINUE command causes a task that executed a PAUSE (SVC 2 code 1) or that was paused by the operator to resume operation. For the Model 3200MPS System, the CONTINUE command causes operation to resume on the task's logical processor.

Format:

CONTINUE [address]

Parameter:

address   is a hexadecimal number that specifies the address at which the task is to resume operation. If this parameter is not specified, the task resumes at the next sequential instruction following the pause.

Functional Details:

For the Model 3200MPS System, the CONTINUE command also applies to tasks paused on an APU, in which case, the task is returned to the CPU for the duration of the pause. When the task is continued and if it is LPU directed, it will be redispatched to the logical processor on the next scheduling event.

Messages:

PARM-ERR

indicates an invalid parameter.

SEQ-ERR

indicates that task was not paused.

TASK-ERR

indicates that there is no currently selected task.

```
------------
|  DELETE   |
------------
```

3.13 DELETE COMMAND

The DELETE command is used to delete direct access files.

Format:

DELETE fd$_1$ $[$,fd$_2$,...,fd$_n]$

Parameter:

fd   identifies the file(s) to be deleted.

Functional Details:

The file being deleted must not be currently assigned to any lu of any task. The DELETE command is not recognized if there are no direct access devices in the system. A file can be deleted only if its write and read protection keys are 0 (X'0000'). If the keys are nonzero, they can be changed using the REPROTECT command.

Messages:

DELE-ERR

indicates that delete failed for reason denoted by TYPE field. See Appendix B for possible entries in the TYPE field.

FD-ERR

indicates an invalid fd.

NODA-ERR

indicates that direct access support is not included in this system.

NOPR-ERR

indicates that no operand was specified.

PRIV-ERR

write or read protection keys are nonzero indicating that the
file is currently assigned.

```
 ------------
| DISPLAY    |
| ACCOUNTING |
 ------------
```

3.14 DISPLAY ACCOUNTING COMMAND

The DISPLAY ACCOUNTING command displays to the specified fd all
accounting information for the specified task.

Format:

$$\text{DISPLAY ACCOUNTING } \left[,\left\{ \begin{array}{c} \text{taskid} \\ \text{all tasks} \end{array} \right\}\right] \left[\left\{ \begin{array}{c} \text{fd} \\ \text{system console} \end{array} \right\}\right]$$

Parameters:

taskid   is the task name for which accounting
                information is to be displayed.   If this
                parameter is omitted, accounting information
                is logged for all tasks.

fd   is the device or file to which accounting
                information is displayed. If this parameter
                is omitted, the accounting information is
                displayed to the system console.

Functional Details:

This command is valid only for systems with accounting support.
For the Model 3200MPS System, tasks that execute on an APU are
not supported by the accounting facility. Therefore, accounting
information for these tasks is neither compiled nor reported when
accounting information is summarized. For other tasks, the time
is displayed in the following format:

hh:mm:ss.t

Where:

hh   represents hours

mm   represents minutes

ss   represents seconds

t   represents tenths of a second

Example:

```
D A ,RCORDEL
TASK NAME USER TIME SVC TIME WAIT TIME ROLL TIME ROLLS I/O
RCORDEL    3:31.60    3.70     45.40     0.0      0  4390
```

Messages:

ASGN-ERR TYPE=

indicates that output file or device could not be assigned for the reason noted in the TYPE field. See Appendix B for possible entries in the TYPE field.

FD-ERR

indicates an invalid fd.

FORM-ERR

indicates a command syntax error.

I/O-ERR

indicates that I/O error was encountered on output file or device.

NOAC-ERR

indicates that accounting is not supported for this system.

NO ACTIVE TASK(S) FOUND

indicates no active tasks found in the system.

PARM-ERR

indicates an operand syntax error.

SPAC-ERR

indicates that required system space is not available.

```
 ------------
|  DISPLAY   |
|  DEVICES   |
 ------------
```

3.15 DISPLAY DEVICES COMMAND

The DISPLAY DEVICES command allows the operator to determine the
physical address, keys, online/offline state, and the volume name
(for online direct access devices) of all devices in the system;
to determine the state of error recording for those devices
supporting the error recording function; and to obtain the names
of the pseudo devices (created by the system tasks using the
intercept feature) currently in the system.

Format:

$$\text{DISPLAY DEVICES} \left[\left\{\begin{matrix} fd \\ \text{system console} \end{matrix}\right\}\right]$$

Parameter:

fd   is the file descriptor specifying the file or
                            device to which the display is routed. If fd
                            is omitted, the display is output to the
                            system console.

Example:

```
    D D
    NAME    DN KEYS
    NULL     0 0000
    CON      2 0000
    CR       4 0000
    PRT     62 0000
    PTRP    13 0000
    PR       0 0000       SPOL
    SPL      0 0000       SPOL
    VDU1    30 0000
    VDU2    14 0000       ITAM
    MAG1    85 0000
    DSC1    C6 0000       MUD1     PROT     CDIR
    DSC2    C7 0000       FIX2
    DSC3    D6 0000       MTM      PROT     ERC
    DSC4    D7 0000       FIX4     ERR=10
    DSC5    E6 0000       OFF
    D67A    FC 0000       V67A     CDIR     OVFL
    VDEV:********.***
      *
```

In the DISPLAY DEVICES output, columns 1, 2, and 3 contain the
device name, device number (address), and keys, respectively.
Column 4 is defined only for pseudo print (spool), data
communications, and direct access devices. The characters SPOL
specify that a device is a pseudo print device used in spooling.
The characters ITAM specify that a device is a data communication
device.

For direct access devices, column 4 contains the characters OFF
to indicate that a device is offline. If the device is online,
the volume name is output in column 4. For write-protected
disks, column 5 contains the characters PROT. If the secondary
directory option is enabled, column 6 contains the characters
CDIR. If the secondary directory is in an overflow state, OVFL
is also displayed.

Error recording information for devices where error recording is
enabled is displayed in columns 5 and 6. ERC indicates that
error recording is enabled, but no errors have been detected.
ERR=n, where n is the number of errors detected, indicates that
errors have been detected. The value of n represents the total
number of errors recorded since the device was last marked on.

The output, VDEV:, is a virtual display device. The asterisks
indicate that a generic pseudo device was created by an SVC
intercept.

Messages:

ASGN-ERR

    indicates an optional fd could not be assigned; e.g., fd is
    assigned for exclusive use already.

FD-ERR

    indicates an invalid fd.

FORM-ERR

    indicates a command syntax error.

IO-ERR

    indicates an I/O error was encountered on output device or
    file.

PARM-ERR

    indicates an operand syntax error.

```
 ------------
|  DISPLAY   |
|  DFLOAT    |
 ------------
```

3.16 DISPLAY DFLOAT COMMAND

The DISPLAY DFLOAT command displays to the specified fd the
contents of the double precision floating point registers
associated with the currently selected task.

Format:

$$\underline{D}ISPLAY \ \underline{DF}LOAT \ \left[, \left\{ \begin{array}{c} fd \\ \text{system console} \end{array} \right\} \right]$$

Parameter:

fd   is the file descriptor specifying the file or
                      device to which the contents of the double
                      precision floating point registers associated
                      with the currently selected task are
                      displayed. If fd is omitted, the display is
                      output to the system console.

Functional Details:

The selected task should have been built with the DFLOAT option
at Link time.

For Model 3200MPS Systems, the APU associated with the task can
be halted to ensure that valid data is recorded.

Example:


```
    D DFL
    PSW  000077F0    0000E588
    0,2  00000000    00000000    00000000    00000000
    4,6  00000000    00000000    00000000    00000000
    8,A  00000000    00000000    00000000    00000000
    C,E  00000000    00000000    00000000    00000000
```

Messages:

ASGN-ERR

    indicates that optional fd could not be assigned; e.g., fd is assigned for exclusive use already.

FD-ERR

    indicates an invalid fd.

FORM-ERR

    indicates a command syntax error.

IO-ERR

    indicates that an I/O error was encountered on output file or device.

NOFP-ERR

    indicates that specified task was not established with the DFLOAT option at Link time.

PARM-ERR

    indicates an operand syntax error.

TASK-ERR

    indicates that no task was specified or the specified task does not exist.

For the Model 3200MPS System, the following message can be displayed:

APU-ERR

    indicates that the task is being actively executed on the APU and that the APU is being controlled by a task other than the operating system command processor.

```
-----------
|  DISPLAY  |
|  ERRORS   |
-----------
```

3.17 DISPLAY ERRORS COMMAND

The DISPLAY ERRORS command displays to the user-specified fd a
list of errors that the hardware error logger has detected, and
the following additional error recording status:

● Status of the error recording function (on or off)

● Date when error recording was turned on (month, day, year)

● Time (hour, minutes, seconds)

● Setting of the error log readout time period (minutes)

Format:

$$\text{DISPLAY ERRORS} \quad \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{system console} \end{array} \right\} \right]$$

Parameter:

fd   is an optional file descriptor specifying the
                            file or device that receives the summary of
                            the hardware logged errors. If fd is omitted,
                            the summary is displayed to the system
                            console.

Examples:

```
ERROR RECORDING STATUS REPORT:
RECORDING STATUS:   OFF
MEMORY ERRORS RECORDED:              0
DISK ERRORS RECORDED:                0
FILE MANAGER ERRORS RECORDED:        0
SYSTEM DETECTED ERRORS RECORDED:     0
SYSTEM MILESTONES RECORDED:          0
MEMORY ERRORLOG READOUT PERIOD:      10
```

```
ERROR RECORDING STATUS REPORT:
RECORDING STATUS:   ON
START DATE/TIME:  05/15/79  14:57:10
CURRENT ERROR LOG FILE:  FIXED:ERR.LOG
MEMORY ERRORS RECORDED:          0
DISK ERRORS RECORDED:            2
FILE MANAGER ERRORS RECORDED:    1
SYSTEM DETECTED ERRORS RECORDED: 0
SYSTEM MILESTONES RECORDED:      15
MEMORY ERRORLOG READOUT PERIOD:  10
```

Functional Details:

The error counts displayed when a DISPLAY ERRORS command is issued represent the total number of errors recorded since the error logging facility was last turned on. Turning the error logging facility off causes the counters to be reset to zero.

Messages:

ASGN-ERR

indicates that an optional fd could not be assigned; e.g., fd is assigned for exclusive use already.

ERRC-ERR

indicates that error recording was not specified at sysgen time.

FD-ERR

indicates an invalid fd.

FORM-ERR

indicates a command syntax error.

IO-ERR

indicates that an I/O error was encountered on output file or device.

PARM-ERR

indicates an operand syntax error.

```
 ------------
|  DISPLAY   |
|   FILES    |
 ------------
```

3.18 DISPLAY FILES COMMAND

The DISPLAY FILES command permits information from the directory of one or more direct access files to be output to the system console or, optionally, to a named file or device.

Format:

$$
\text{DISPLAY FILES} \quad \left[\begin{array}{c} \left[, \left\{ \begin{array}{c} \text{voln:} \\ \text{default sys vol} \end{array} \right\} \right] \left[\left\{ \begin{array}{c} \text{filename} \\ * \\ - \end{array} \right\} \right] \left[. \left[\left\{ \begin{array}{c} \text{ext} \\ - \end{array} \right\} \right] \right] \\ \left[/ \left\{ \begin{array}{c} \text{actno} \\ - \\ 0 \\ S \end{array} \right\} \right] \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{sys console} \end{array} \right\} \right] \end{array} \right]
$$
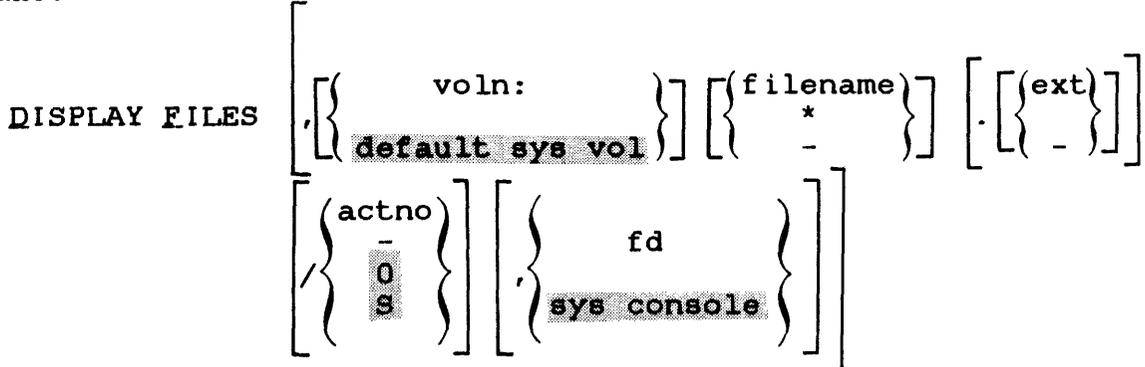
Parameters:

voln: is a 1- to 4-character name of a disk volume. The first character must be alphabetic, the remaining alphanumeric. If voln is omitted, the default system volume is assumed.

filename   is a 1- to 8-character name of a file. The first character must be alphabetic, the remaining alphanumeric.

.ext   is a 1- to 3-character extension to the filename.

actno   is a decimal number from 0 through 65,535 specifying the account number associated with the file. The account number field may be omitted, in which case system files (account 0) are displayed.

fd   is an optional file descriptor specifying the file or the device to which the display is output. If fd is omitted, the display is output to the system console.

Functional Details:

A hyphen (-) in the command format requests that all files starting with the characters preceding the - be displayed, subject to any restrictions specified in the extension, account number, and fd fields.

Examples:

CAL32-   displays all files whose first five characters are CAL32.

CAL32.-   displays all files named CAL32 with any extension.

A hyphen can also be used in a command format to request that all files with the extension following the hyphen be displayed.

Example:

-.OBJ   displays all files with the extension .OBJ.

The character * requests that all files with matching characters in the same position(s) as those entered are displayed.

Examples:

CAL32***   displays all files between five and eight characters in length whose first five characters are CAL32.

CAL**CAL   displays all files whose first three and last three characters are CAL.

****32.OBJ   displays all files with a filename containing six characters whose fifth and sixth characters are 32 and whose extension is .OBJ.

The characters * and - can be combined in the command format, as previously described, to further delimit displayed files.

Example:

CAL**1-   displays all files whose first three
                           characters are CAL, and whose sixth
                           character is 1.

An example of the display produced by the DISPLAY FILES command
is:


```
    D F

        FILENAME......     TY DBS/IBS RECL. RECORDS CREATED....... LAST WRITTEN.. KEYS
        DISCLAIM.038/00000 IN  5/1     80      35  2/15/82 09:57  2/15/82 09:57 0000
        EDITINFO.32 /00000 IN  5/1     80      58  1/27/82 14:34  1/27/82 14:34 0000
        PRECAL  .640/00000 IN  5/1     80      84 10/29/81 14:11 10/29/81 14:11 0000
        APCMAC  .057/00000 IN  5/1     80     364  3/17/82 13:54  3/17/82 13:54 0000
        CONVOY  .DLK/00000 IN  5/1     80      19  5/20/82 09:54  2/09/83 11:16 0000
        CROW2   .CMD/00000 IN  1/1    120       7  2/07/83 17:53  2/07/83 17:53 0000
        APBFORT .627/00000 IN  5/1     80      52 12/04/80 08:57 12/04/80 08:57 0000
        CH2FB   .063/00000 IN  5/1     80     467  2/22/83 11:22  2/22/83 11:22 0000
        APCTAM  .541/00000 IN  5/1     80      47 12/15/80 13:23  2/19/81 17:25 0000
        APATAM  .541/00000 IN  5/1     80      23 12/15/80 13:21  2/19/81 17:25 0000
        INDINP48.086/00000 IN  5/1     80     244  9/22/82 16:56  9/22/82 16:56 0000
        APCEM2  .012/00000 IN  5/1     80      81  2/18/80 12:42  2/18/80 12:42 0000
        TITCIM  .091/00000 IN  5/1     80      23  6/17/82 09:34  6/17/82 09:48 0000
        APBRMA  .579/00000 IN  5/1     80     297 11/29/79 15:39 11/29/79 15:39 0000
        MANUALS .SR /00000 IN  5/1     80     183  5/13/82 09:50  5/13/82 09:50 0000
        DISCLAIM.478/00000 IN  5/1     80      35 11/13/80 10:50 11/13/80 10:50 0000
        CH3FORT .627/00000 IN  5/1     80     221  3/21/80 08:34  3/21/80 08:34 0000
        CH1RMA  .579/00000 IN  1/1     80     221  5/06/80 13:48  5/06/80 13:49 0000
        TITLE   .466/00000 IN  5/1     80      21  7/09/80 16:44 12/12/80 15:49 0000
        CH1AID  .374/00000 IN  5/1     80     299  2/08/80 09:42  2/08/80 09:42 0000
        APAFLD  .020/00000 IN  5/1     80      91  3/13/80 09:45  3/13/80 09:45 0000
        SORT    .TSK/00000 CO              81  7/20/79 16:31  7/20/79 16:32 0000
        KIRM    .JPC/00000 EC 64/3         100  2/25/83 11:02  2/25/83 11:02 0000
        KIRMA   .JPC/00000 NB 64/3    126   78  2/25/83 11:04  2/25/83 11:04 0000
```


For contiguous files, TYPE (TY) is CO, and RECORDS is the size of
the file in (decimal) sectors.

For indexed files, TYPE is IN, followed by the data and index
blocking factors, RECL is the logical record length in (decimal)
bytes, and RECORDS is the number of logical records (in decimal)
in the file.

For nonbuffered indexed files, TYPE is NB, RECL is the logical
record length in (decimal) bytes, and RECORDS is the number of
logical records (in decimal) in the file.

For extendable contiguous files, TYPE is EC, and RECORDS is the
length of the file in sectors (i.e., the size of the file).

Spool and temporary files are named as *SPOOLFILE* and *TEMPFILE* respectively.

For all files, ACT is the associated user account number. For OS/32 systems containing multi-terminal monitor (MTM), the account number denotes the unique user to whom the file belongs. See the OS/32 Multi-Terminal Monitor (MTM) Reference Manual for further information.

Examples:

D F	displays to the console device all files with account number 0 on the default system volume.
D F,CAL32.TSK/-	displays file CAL32.TSK in any account.
D F,-/-	displays all files on the disk.
D F,,MAG1:	displays to the device MAG1 all files with account number 0 on the default system volume.
D F,M300:	displays to the console device all files with account number 0 on volume M300.
D F,M300:A-.TSK	displays all files on volume M300 with first character A and extension TSK in account number 0.
D F,-.,PR1:	displays all files on the default system volume in account number 0 with blank extension, regardless of filename. The display is routed to device PR1.
D F,CAL**1-.-	displays, to the console device, all files that start with CAL, contain the character 1 in the sixth position, have any extension, and are in account number 0.
D F,:CAL32.TSK	displays all files with the filename, CAL32.TSK, on all volumes.

Messages:

ASGN-ERR

    indicates that an optional fd could not be assigned; e.g., fd is assigned for exclusive use already.

FD-ERR

    indicates an invalid optional fd.

FILE NOT FOUND

    indicates that the specified file was not found.

FORM-ERR

    indicates a command syntax error.

IO-ERR

    indicates that an I/O error was encountered on output device or file.

NODA-ERR

    indicates that direct access support is not included in this system.

NO DIRECTORY ENTRIES ON voln

    indicates that voln has no files on it.

3.19 DISPLAY FLOAT COMMAND

The DISPLAY FLOAT command displays to the specified fd the
contents of the single precision floating point registers
associated with a user-specified task.

Format:

$$\text{DISPLAY FLOAT} \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{system console} \end{array} \right\} \right]$$

Parameter:

fd   is an optional file descriptor specifying the
file or device to which the display is output.
If fd is omitted, the display is output to the
system console.

Functional Details:

The user-specified task must be established with the FLOAT option
specified at Link time.

For the Model 3200MPS System, the APU associated with the task
can be halted to ensure that valid data is recorded.

Example:

```
D FL
PSW   000077F0     0000E588
0,2   00000000     00000000
4,6   00000000     00000000
8,A   00000000     00000000
C,E   00000000     00000000
```

Messages:

ASGN-ERR

    indicates that an optional fd could not be assigned; e.g., fd already is assigned for exclusive use.

FD-ERR

    indicates an invalid fd.

FORM-ERR

    indicates a command syntax error.

IO-ERR

    indicates that an I/O error was encountered on output file or device.

NOFP-ERR

    indicates that specified task was not established with the FLOAT option at Link time.

PARM-ERR

    indicates an operand syntax error.

TASK-ERR

    indicates that no task was specified or user-specified task does not exist.

For the Model 3200MPS System, this message can be displayed:

APU-ERR

    indicates that the task is APU active and that the APU is being controlled by a task other than the operating system command processor.

3.20 DISPLAY ITAMTERM COMMAND

The DISPLAY ITAMTERM command permits information relative to allocated data communications LCBs to be output to the system console or, optionally, to a named file or device.

Format:

$$
\text{DISPLAY ITAMTERM ,voln:} \left[\begin{Bmatrix} \text{filename} \\ - \end{Bmatrix} \right] \left[. \left[\begin{Bmatrix} \text{ext} \\ - \end{Bmatrix} \right] \right]
$$

$$
\left[, \begin{Bmatrix} \text{fd} \\ \text{sys console} \end{Bmatrix} \right]
$$

Parameters:

voln:	is the data communications device to be searched.
filename	is a 1- to 8-character name. The first character must be alphabetic, the remaining alphanumeric.
.ext	is a 1- to 3-character extension.
fd	is the file descriptor of the file or device on which the display is to be output. If fd is omitted, the display is output to the system console.

The information displayed is:

● Device name, filename, and extension

● Device code (decimal)

● Data communications extended device code (hexadecimal)

● Transmission block size (decimal)

● Logical record size (decimal)

● Number of transmission blocks (decimal)

Examples:

D I,BSCI:	displays all data communications terminal LCBs with device mnemonic BSCI:.
D I,BSCI:-.-	displays all data communications terminal LCBs with device mnemonic BSCI: and any extension.
D I,BSCI:-.LIN	displays all data communications terminal LCBs with device mnemonic BSCI: and extension LIN.
D I,BSCI:AJM.LIN	displays all data communications terminal LCBs with device mnemonic BSCI, filename AJM, and extension LIN.
D I,BSCI:,MAG1:	displays on device MAG1:, all data communications terminal LCBs with device mnemonic BSCI:.

Messages:

ASGN-ERR

indicates that an optional fd could not be assigned; e.g., fd is assigned for exclusive use already.

FD-ERR

indicates an invalid optional fd.

FORM-ERR

indicates a command syntax error.

IO-ERR

indicates that an I/O error was encountered on output device.

NOPR-ERR

indicates that a required operand is missing.

indicates an operand syntax error or data communications devices not supported in the system.

SPECIFIED TERMINAL(S) NON-EXISTENT FOR voln

indicates that a specific data communications terminal is not found or voln has no LCBs allocated for it.

```
------------
| DISPLAY LU |
------------
```

3.21 DISPLAY LU COMMAND

The DISPLAY LU command permits the operator to display all
assigned logical units of the currently selected task.

Format:

$$\text{DISPLAY LU} \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{sys console} \end{array} \right\} \right]$$

Parameter:

fd   is an optional file descriptor specifying the
                        file or device on which the assigned logical
                        units are to be displayed. If fd is omitted,
                        the display is output to the system console.

Functional Details:

The lu number, file or device name, current access privileges,
current record number, and percentage through file are displayed.
The current record number and percentage through file are
displayed only for files.

LU	FILE/DEVICE	RECORD	THRU
1	M67A:RADPROC.CSS/000,SRO	30	15.0%
3	CON:,SRW		
5	CON:,SRW		
6	CON:,SRW		

LU	FILE/DEVICE	RECORD	THRU
1	M67A:RADPROC.CSS/000,SRO	200	100.0%
3	CON:,SRW		
4	M67A:&2614586.001/000,SREW	1	100.0%
5	CON:,SRW		
6	CON:,SRW		

Example:

DISP LU,PR: displays assigned logical units to
                        the printer device (PR:).

Messages:

ASGN-ERR

    indicates that optional fd could not be assigned; e.g., fd is
    assigned for exclusive use already.

FD-ERR

    indicates an invalid optional fd.

FORM-ERR

    indicates a command syntax error.

IO-ERR

    indicates that an I/O error was detected on output device or
    file.

PARM-ERR

    indicates an operand syntax error.

TASK-ERR

    indicates that there is no currently selected task.

```
------------
|  DISPLAY  |
|    MAP    |
------------
```

3.22 DISPLAY MAP COMMAND

The DISPLAY MAP command causes a memory map to be output to the console or to a specified file or device. The display map can be of the entire system or of a particular task, pure, library, or task common segment, system space, pseudo task, or marked off memory.

Format:

$$
\text{DISPLAY MAP} \left[\bigg/ \left\{ \begin{array}{l} \text{id} \\ \text{.TSK} \\ \text{.PUR} \\ \text{.SEG} \\ \text{.SYS} \\ \text{.OFF} \\ \text{.PST} \\ \text{local memory} \end{array} \right\} \right] \text{[/rname]} \left[, \left\{ \begin{array}{l} \text{fd} \\ \text{sys console} \end{array} \right\} \right]
$$
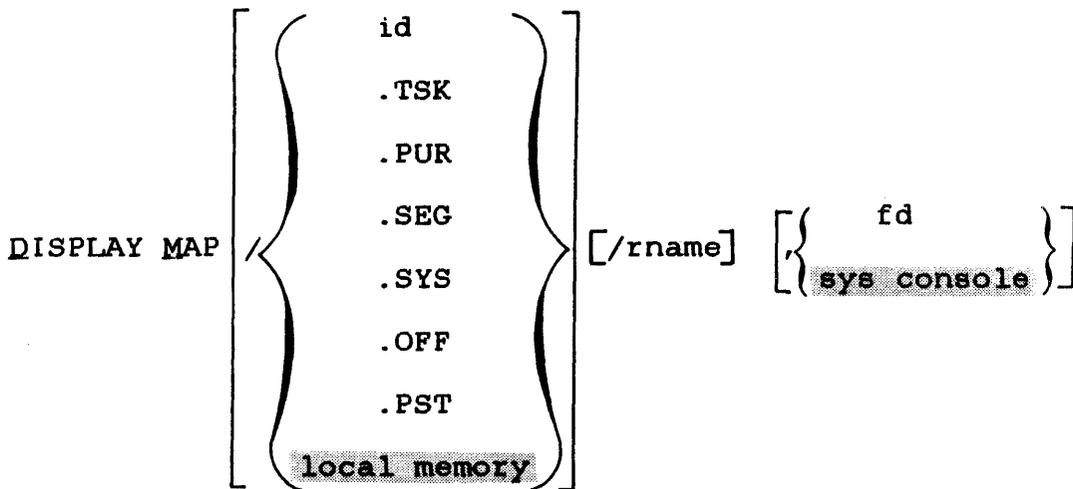
Parameters:

id	can be either .BG to refer to the background task or the name associated with a task.
.TSK	requests display of task segments.
.PUR	requests display of pure segments.
.SEG	requests display of library and task common segments.
.SYS	requests display of system space.
.OFF	requests display of all OFF memory segments.

.PST	requests display of pseudo tasks created by a system task using the intercept feature.
rname	requests display of a specific library segment, task common segment, sharable pure segment, or a taskid.
fd	is the file descriptor specifying the file or device on which the map is to be output; if fd is omitted, the display is output to the system console.

Functional Details:

If segment type and rname are omitted, all tasks, task common, library and pure segments, marked off memory, pseudo tasks, and system space are displayed. The name, segment type, start address, segment size, size of system space, number of shared segments, status and priority are displayed. If rname is specified, only the information for that task or segment is displayed. For id, all sharable segments are also displayed. The first line of the display indicates the starting address and size of the total task memory space.

Examples:

This example shows the display for two system tasks, two foreground tasks, a background task, and two task common segments:

```
D M
      NAME        TYPE   START    SIZE   SEG  SYS  STAT    PRI
  TASK MEMORY             1D000  321.00
  .SPL            .TSK    1D000   11.25                A   128
  .MTM            .TSK    1FD00   33.00           RES  A   128
  USER1           .TSK    28100    4.00   2.29        D   129
  USER2           .TSK    29100   25.75   1.95        D   129*
  .BG             .TSK    2F800    4.00   1.25        D   128
  TSKCOM1         .SEG    30800    2.50
  TSKCOM2         .SEG    31200   10.00
  SYSTEM SPACE            6D400   75.00   8.34
  *
```

The following example shows the display for one sharable pure
segment, four library segments, one task common segment, one off
memory segment, and five foreground task segments.

NAME	TYPE	START	SIZE	SEG	SYS	STAT	PRI
TASK MEMORY		22000	1044.00				
M67AEDIT32	TSK .PUR	24800	28.00	1			
PLUSRTL	.SEG	2B800	2.00	2			
MINUSRTL	.SEG	2C000	2.00	2			
MULRTL	.SEG	2C800	2.00	2			
DIVIDRTL	.SEG	2D000	2.00	2			
ICOM	.SEG	2E000	2.00	2			
OFF MEMORY		C0000	256.00			OFF	
EDIT32	.TSK	22000	10.00	1	2.25	P	128
TEST1	.TSK	2D800	2.00	4	1.25	D	128
TCOMTEST	.TSK	2E800	24.00	1	1.00	D	128
TEST1IN	.TSK	34800	2.00	4	1.25	D	128
LOOP	.TSK	35000	4.00	1	1.00	A	130
SYSTEM SPACE		127000	100.00		12.75		

The NAME field is the name of a task or segment, and .BG
indicates the background segment. Any other name indicates a
foreground task, global task common segment, or library segment.
For pure segments, NAME is the fd of the file containing the pure
segment.

The TYPE field describes the segment type. Possible values are:

.TSK   Task memory image form (impure)

.PUR   Task memory image form (pure)

.SEG   Task common or library

.PST   Pseudo task

The START field is the absolute starting address of the segment
in memory. The system space line contains the starting address
of the system space and represents the upper limit of the task
space plus 1. The task memory line contains the starting address
of the task space and represents the upper limit of the operating
system space plus 1.

The SIZE field indicates the size in kb of the segment. Each
size is a multiple of 0.25. The task memory and system space
lines contain the maximum available task and system space.

The SEG field contains the number of shared segments for a task. For a shared segment, the SEG field indicates the number of tasks currently sharing the segment.

The SYS field indicates the amount of system space a task is using. In the system space line, this field contains the total amount of system space currently in use.

The STAT field indicates the status of these tasks, as follows:

D   Dormant

P   Paused (console wait)

A   Active (any state other than dormant or paused)

The status can be preceded by RES indicating the task is memory resident, ROL indicating the task is rollable, or ROLLED indicating the task is rolled. A task displayed on the map as active may in fact be in a wait state. The system operator can use the TASK and DISPLAY PARAMETERS commands to get the actual wait status halfword for a given task. The PRI field indicates the priority in decimal of all tasks currently in the system. An asterisk (*) next to a value in the priority field indicates that the actual run priority of that task has been lowered by MTM. OFF identifies marked off memory.

Messages:

ASGN-ERR

    indicates that the output device or file could not be assigned; e.g., device returns unrecoverable error on I/O attempt.

FD-ERR

    indicates that an invalid fd was specified.

FORM-ERR

    indicates a command syntax error.

IO-ERR

indicates that an I/O error occurred on output device or file; e.g., device returns unrecoverable error on I/O attempt.

PARM-ERR

indicates an operand syntax error.

SEGMENT REQUEST NOT FOUND

indicates that the requested segment was not found, or no tasks exist in memory.

3.23 DISPLAY PARAMETERS COMMAND

The DISPLAY PARAMETERS command is used to display parameters pertinent to the currently selected task. The display appears on the console device, or on a device or file selected by the operator.

Format:

$$\text{DISPLAY PARAMETERS} \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{sys console} \end{array} \right\} \right]$$

Parameter:

fd   is an optional file descriptor specifying the file or device to which the display is to be output. If fd is omitted, the display appears on the system console.

Functional Details:

Table 3-3 lists the field addresses and data displayed when the DISPLAY PARAMETERS command is entered.

TABLE 3-3   FIELDS DISPLAYED BY THE DISPLAY
PARAMETERS COMMAND

TASK	TASKID	TASK NAME
======	========	===========
CTSW	xxxxxxx	Status portion of current task status word (TSW)
PSW	xxxxxx	Least-significant three bytes of status portion
CLOC	xxxxxx	Current location
STAT	xxxxx	Task wait status
TOPT	xxxxx	Task options
USSP	xxxxx	Current used system space
MUSP	xxxxx	Maximum used system space
MXSP	xxxxx	Maximum allowed system space
CTOP	xxxxx	Task CTOP
UTOP	xxxxx	Task UTOP
UBOT	xxxxx	Task UBOT
SLOC	xxx	Task starting location
NLU	xx	Number of logical units (decimal)
MPRI	xxx	Maximum priority (decimal)
SVOL	xxxx	Default volume id
TVOL	xxxx	Current task volume

The addresses displayed as CTOP, UTOP, UBOT, and SLOC are not physical addresses, but are addresses within the task's own program space. CLOC may be a program space address or a physical address in a system subroutine being executed on behalf of the task. NLU is given in decimal. SVOL is the ASCII system volume id. It is not specifically related to the currently selected task, but is given here for operator convenience. TVOL is the ASCII task volume name associated with the currently selected task. CTSW is given in hexadecimal. For a definition of the status portion of the TSW, and a description of the fields CTOP, UTOP, UBOT, and SLOC, see the OS/32 Application Level Programmer Reference Manual.

TOPT is given in hexadecimal. The definitions of task option bits are listed in Table 3-4.

TABLE 3-4   TASK OPTION BIT DEFINITIONS

BIT	MASK	MEANING
0	8000 0000	0 = task type determined by bit 16 1 = task is a d-task
*1	4000 0000	0 = task can run on CPU or APU 1 = task cannot run on CPU
*2	2000 0000	0 = no APU mapping allowed 1 = task can perform APU mapping functions
*3	1000 0000	0 = no APU control allowed 1 = task can perform APU control functions
*4	0800 0000	0 = dynamic priority scheduling disabled 1 = dynamic priority scheduling enabled
5	0400 0000	0 - prompts disabled 1 - prompts enabled
6	0200 0000	0 - I/O interpreted without VFC (except where specified) 1 - all I/O interpreted with VFC
7	0100 0000	0 - no extended SVC 1 parameter blocks used (excludes communications I/O) 1 - extended SVC 1 parameter blocks used
8	0080 0000	0 - new TSW for task event service 1 - no new TSW for task event service
9	0040 0000	0 - task event all registers saved 1 - task event partial register saved
10	0020 0000	0 - task event no register saved 1 - task event register saved
11	0010 0000	0 - not in system group 1 - in system group
12	0008 0000	0 - no console I/O intercept 1 - console I/O intercept enable (MTM)

TABLE 3-4   TASK OPTION BIT DEFINITIONS (Continued)

BIT	MASK	MEANING
13	0004 0000	0 - universal status reports not allowed 1 - universal status reports allowed
14	0002 0000	0 - allow e-task load 1 - prevent e-task load
15	0001 0000	0 - queued I/Os not purged on error 1 - queued I/Os purged on error
16	0000 8000	0 - u-task 1 - e-task
17	0000 4000	0 - task will pause due to arithmetic fault 1 - task does not pause due to arithmetic fault
18	0000 2000	0 - no single precision floating point 1 - single precision floating point
19	0000 1000	0 - task will be removed from memory at end of task 1 - task remains in memory after end of task
20	0000 0800	0 - allow SVC 6 control call 1 - prevent SVC 6 control call
21	0000 0400	0 - allow SVC 6 communication call 1 - prevent SVC 6 communication call
22	0000 0200	0 - background task pauses upon SVC 6 communication request 1 - requests for communication calls by background task are ignored
23	0000 0100	0 - no double precision floating point 1 - double precision floating point
24	0000 0080	0 - task is not rollable 1 - task is rollable
25	0000 0040	0 - no overlay 1 - use overlay
26	0000 0020	0 - accounting disabled 1 - accounting enabled
27	0000 0010	0 - task can issue intercept calls 1 - task cannot issue intercept calls

TABLE 3-4 TASK OPTION BIT DEFINITIONS (Continued)

BIT	MASK	MEANING
28	0000 0008	0 - no account privileges 1 - file account privileges
29	0000 0004	0 - bare disk assign not allowed 1 - bare disk assign allowed
30	0000 0002	0 - not universal 1 - universal
31	0000 0001	0 - no keychecks 1 - do keychecks

* Bits 1 through 4 are the task option bits applicable
  only to the Model 3200MPS System.

STAT is given in hexadecimal. The definitions of wait status
bits are shown in Table 3-5.

TABLE 3-5 WAIT STATUS BIT DEFINITIONS

BIT	MASK	MEANING
15	0001 0000	Intercept wait
16	0000 8000	I/O wait
17	0000 4000	(Any) IOB/WAIT
18	0000 2000	Console wait (paused)
19	0000 1000	Load wait
20	0000 0800	Dormant
21	0000 0400	Trap wait
22	0000 0200	Time of day wait
23	0000 0100	Suspended
24	0000 0080	Interval wait
25	0000 0040	Terminal wait

TABLE 3-5   WAIT STATUS BIT DEFINITIONS (Continued)

BIT	MASK	MEANING
26	0000 0020	Roll pending wait
27	0000 0010	Intercept initialization (MTM)
28	0000 0008	Intercept termination (MTM)
29	0000 0004	System resource connection wait
30	0000 0002	Accounting wait

NOTE

Zero status indicates the task is active.

| **Functional Details:**

| For Model 3200MPS Systems, the APU associated with the task can
| be halted to ensure that valid data is displayed.

Example:

The following is an example of the output generated in response
to a DISPLAY PARAMETERS command:

```
D P
TASK      EDIT32
CTSW      00001000
PSW          477F0
CLOC         F2B7C
STAT          2000
TOPT         10021
USSP          14F8
MUSP          2208
MXSP          3000
CTOP          24FE
UTOP          2370
UBOT             0
SLOC         F0000
NLU             15
MPRI           128
SVOL          M67A
TVOL          M67A
```

Messages:

ASGN-ERR

     indicates that an optional fd could not be assigned; e.g., fd is already assigned for exclusive use.

FD-ERR

     indicates an invalid fd.

FORM-ERR

     indicates a command syntax error.

IO-ERR

     indicates that an I/O error was detected on output device or file.

PARM-ERR

     indicates an operand syntax error.

TASK-ERR

     indicates that there is no currently selected task.

For the Model 3200MPS System, the following message can also be displayed:

APU-ERR

     indicates that the task is APU active and that the APU is being controlled by a task other than the operating system command processor.

```
------------
|  DISPLAY   |
|  REGISTERS |
------------
```

3.24 DISPLAY REGISTERS COMMAND

The DISPLAY REGISTERS command displays to the specified fd the contents of the general purpose user registers associated with a user-specified task.

Format:

    DISPLAY REGISTERS [,fd]

Parameter:

    fd   is the file descriptor to which the contents
                          of the general purpose user registers are
                          displayed.

| Functional Details:

| For Model 3200MPS Systems, the APU associated with the task may
| be halted to ensure that valid data is displayed.

Example:

    D R
    PSW   000077F0   0000E588
    0-3   00000000   00000000   00000000   00004801
    4-7   0000E83C   00000000   00000000   0000D2EA
    8-B   0000E8CB   00000000   0000E848   00000028
    C-F   0000E804   0000E9D0   0000E584   0000E05E

Messages:

ASGN-ERR

    indicates that an optional fd could not be assigned; e.g., fd
    is already assigned for exclusive use.

FD-ERR

    indicates an invalid fd.

FORM-ERR

    indicates a command syntax error.

IO-ERR

    indicates that an I/O error was detected on output device or
    file.

PARM-ERR

    indicates an operand syntax error.

TASK-ERR

    indicates that no task was specified or user-specified task
    does not exist.

For Model 3200MPS Systems, the following message may appear:

APU-ERR

    indicates that the task is APU active and that the APU is
    being controlled by a task other than the operating system
    command processor.

```
-----------
|  DISPLAY  |
|   TASKS   |
-----------
```

| 3.25 DISPLAY TASKS COMMAND

| This command is supported in revision 7.1 or higher of OS/32.

| The DISPLAY TASKS command outputs status information for all
| tasks in the system or for a single specified task.

| Format:

| $$\text{DISPLAY TASKS [task-id]} \left[, \begin{Bmatrix} \text{fd} \\ \text{sys console} \end{Bmatrix} \right]$$

| Parameters:

| task-id   specifies the task for which status
| information is to be displayed. If this
| parameter is omitted, status information for
| all tasks in the system is displayed.

| fd   specifies the destination for the output
| generated by this command. The destination
| may be a file with account number 0 or another
| character handling device. If fd is omitted,
| the output is directed to the system console.

| Functional Details:

| This command provides the system operator with the name, size,
| memory resources, dispatch priority, and execution state of each
| task in the system.

Examples:

```
*DISPLAY TASKS

TASK-ID  SIZE    SHD  SYS    PRI  LPU#   STATUS
 .SPL    12.00   0           128  0      WAITING I/O, IOB, TRAP
 .MTM    34.00   0           128  0      READY
 EXEC    38.00   1    2.50   129  1-DIR  ACTIVE APU3
 SOAKER  102.25  1    10.25  129  1-DIR  READY APU3

*D TA SOAKER

TASK-ID  SIZE    SHD  SYS    PRI  LPU#   STATUS
 SOAKER  102.25  1    10.25  129  1      READY APU3
```

Fields:

TASK-ID — identifies the task name.

SIZE — is the sum of all segment sizes used by this task in kilobytes.

SHD — is the number of shared segments used by this task.

SYS — is the current amount of system space used by this task in kilobytes.

PRI — is the dispatch priority on the CPU. The lower the number the higher the priority.

LPU# — is the logical processor unit number assigned to the task. The abbreviation DIR identifies an LPU directed task. Zero indicates CPU. This parameter applies to the Model 3200MPS System only.

STATUS — gives the current execution status of the task. The following are valid execution states:

ACTIVE APU# — task active on APU if specified.

READY or READY APU# — task ready to run on CPU or on APU if specified.

WAITING "EVENT LIST" — task is waiting for completion of the specified events. The events defined in the event list can contain one or more of the following mnemonics.

```
I/O       (input/output)
IOB       (I/O block)
PAUS      (pause)
SUSP      (suspend)
TRAP      (trap)
INTRCP    (intercept)
INTRVL    (interval)
CONECT    (connection)
ACT       (accounting)
DORM      (dormant)
LOAD      (load)
TOD       (time of day)
TERM      (terminal)
ROLL      (roll)
MTM       (multi-terminal monitor)
```

Messages:

ASGN-ERR

indicates that an optional fd could not be assigned; e.g., fd is already assigned for exclusive use.

FD-ERR

indicates an invalid fd.

FORM-ERR

indicates a command syntax error.

IO--ERR

indicates that an I/O error was detected on output device or file.

PARM-ERR

indicates an operand syntax error.

TASK(S) NOT FOUND

specified task(s) was not found in the system or, in the case of a full display, no tasks were found in the system.

3.26 DISPLAY TIME COMMAND

The DISPLAY TIME command causes the current date and time to be output to the system console or to a specified file or device.

Format:

$$
\text{DISPLAY TIME} \left[, \begin{Bmatrix} \text{fd} \\ \underline{\text{sys console}} \end{Bmatrix} \right]
$$

Parameter:

    fd   specifies the file or device to which the display is to be output. If fd is omitted, the display is output to the system console.

Functional Details:

The display has the following format:

    mm/dd/yy   hh:nn:ss

or alternatively by sysgen option:

    dd/mm/yy   hh:nn:ss

Messages:

ASGN-ERR

    indicates that an optional fd could not be assigned; e.g., fd already is assigned for exclusive use.

FD-ERR

    indicates an invalid fd.

FORM-ERR

    indicates a command syntax error.

IO-ERR

    indicates that an I/O error was encountered on output device or file.

PARM-ERR

    indicates an operand syntax error.

3.27 DISPLAY VOLUME COMMAND

The DISPLAY VOLUME command displays, to the specified fd, the
state of the specified disk volume. This command is supported
only on disk devices that support error recording; e.g., mass
storage media (MSM) disk devices.

Format:

$$\text{DISPLAY } \underline{\text{VOLUME}} \text{ ,voln} \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{system console} \end{array} \right\} \right]$$

Parameters:

voln   is the volume name of the disk whose status is
              displayed.

fd   is an optional file descriptor specifying the
              file or device to which the specified disk
              volume's status is displayed. If fd is
              omitted, the display is output to the system
              console.

Functional Details:

If the disk device for the specified volume name supports error
recording, information is displayed showing read, write, and
total SVC 1 requests (REQUESTS=); the number of sectors read from
or written to and the total number of sectors accessed (SECTOR=);
and the average latency for the SEEK/REQ or ROTATION fields
(AVERAGE LATENCY:).

If the disk device for the specified volume name does not support
error recording, the only information displayed is the amount of
free space on the specified volume (FREESPACE=). See examples
below.

Examples:

| D V,M300

      VOLUME= M300
      FREESPACE = 222498 SECTORS   23959 EXTENTS   22.23% OF TOTAL
      REQUESTS   =   47461 READ   21552 WRITTEN   69013 TOTAL
      SECTORS   = 588784 READ   112624 WRITTEN   701408 TOTAL
      AVERAGE LATENCY: 0.377 SEEK/REQ   33 ROTATION (SECTORS/REQ)

Volume M300 has error recording capabilities.

      D V,FIXD

      VOLUME= FIXD
      FREESPACE =   18676 SECTORS   6 EXTENTS   95.36% OF TOTAL

Volume FIXD has no error recording capabilities.

Fields:

FREESPACE=   is the amount of free space on the specified
                    volume displayed as number of sectors, number
                    of extents, and percentage of total space.

          SECTORS   is the number of free sectors
                          on the specified volume.

          EXTENTS   is the number of contiguous
                          free blocks of sectors on the
                          specified volume.

REQUESTS=   is the number of read, write, and total SVC 1
                    requests made to the specified volume.

SECTORS=   is the number of sectors read from, written
                    to, and total number of sectors accessed.

AVERAGE   in the SEEK/REQ field, the average latency is
LATENCY: the average number of seeks done per I/O
                    request. In the ROTATION field, the average
                    latency is the average wait for disk
                    rotation/request measured in sectors. For MSM
                    disks, one rotation equals 64 sectors.
                    Therefore, half a rotation equals 32 sectors.

Messages:

ASGN-ERR

   indicates the optional fd or disk device of volume name, voln, could not be assigned.

DEVICE NOT A DISC

   indicates that the specified device with volume name, voln, is not a bulk storage device.

FD-ERR

   indicates an invalid optional fd.

FORM-ERR

   indicates a command syntax error.

IO-ERR

   indicates that an I/O error was encountered on disk device of volume name, voln, or on output device.

NODA-ERR

   indicates that direct access support is not included in this system.

NOPR-ERR

   indicates a required operand is missing.

PARM-ERR

   indicates an operand syntax error.

3.28 ERROR LOG COMMAND

The ERROR LOG command turns on or off the error recording function, which copies the errors from the error logger to the disk. This command controls general error recording for all processors and memory error recording for the Perkin-Elmer Series 3200 processors.

Format:

$$\underline{\text{ERROR LOG}} \; , \begin{Bmatrix} \text{ON} & \big[\text{,} [\text{fd}] & [\text{, INIT}] \big] \\ \underline{\text{OFF}} & \end{Bmatrix}$$

Parameters:

OFF   dumps the internal error record buffer to disk and closes the error recording files. Internally, errors are still stored, but not written to disk, which causes error data to be lost if the internal buffer overflows.

ON   turns on the error recording function.

fd   specifies the contiguous file to be used for error recording. If this parameter is omitted, the file specified at sysgen is the default. If the user-specified fd does not currently exist, the file is automatically allocated and assigned.

NOTE

To facilitate easy location of the error recording file by maintenance and support personnel, it is recommended that a standard error recording filename, SYSERROR.LOG, be used and that it be allocated on the default system volume.

INIT   initializes the error recording file so that
                        new   error   records   can   be   added   to   the
                        beginning of the file. If this parameter   is
                        omitted, subsequent error records are added
                        following the last records written to the
                        file.

Functional Details: |

If the date and time parameters of the SET TIME command were not |
specified, the command ERROR LOG ON cannot record valid dates and |
times in the error logger file, and an error message is |
displayed. |

Messages:

CLOS-ERR TYPE=BUFF

    indicates an error occurred when closing the error recording
    file.   TYPE=BUFF means system space has become corrupted and
    buffers and/or file control blocks cannot be returned to the
    free system space.

DATE-ERR

    indicates that the ON parameter was specified but the date
    and time parameters of the SET TIME command were not
    specified.

NOPR-ERR

    indicates that no parameters were specified.

OFF-ERR

    indicates that the OFF parameter was specified but the error
    recording function was already off.

ON-ERR

    indicates that the ERROR LOG command was entered twice with
    the ON parameter specified.

PARM-ERR

    indicates that a syntax error exists in the specified parameter.

PRES-ERR

    indicates that error recording is not supported.

3.29 ERROR PERIOD COMMAND

The ERROR PERIOD command sets the memory error log readout period
to a user-specified number of minutes. The initial value is
specified at sysgen time. This command can be used only with
Perkin-Elmer Series 3200 processors.

Format:

$$\text{ERROR PERIOD} \left[, \left\{ \begin{matrix} \text{minutes} \\ * \end{matrix} \right\} \right]$$

Parameters:

minutes   is a decimal number from 1 through 1440
                    specifying the number of minutes between error
                    log readouts. If this parameter is omitted,
                    the time period is reset to the initial sysgen
                    value.

*   specifies that the memory error logger
                    performs a readout immediately. The
                    previously set readout period is not affected.

Messages:

INIT-ERR

    indicates that memory error recording was initialized but it
    already had been specified and was in progress.

NOPR-ERR

    indicates that no parameters were specified.

PARM-ERR

    indicates that a syntax error exists in the specified
    parameter.

PERD-ERR

indicates that the number of minutes specified for the error log readout period was not a number from 1 through 1440.

SPAC-ERR

indicates that there was not enough system space allocated for an error log buffer.

3.30 ERROR RECORDING COMMAND

The ERROR RECORDING command turns error recording on or off for a specified device.

Format:

$$\text{ERROR RECORDING ,fd,}\left\{\begin{array}{l}\text{ON}\\\text{OFF}\end{array}\right\}$$

Parameters:

fd   is the file descriptor of the device for which errors are being recorded.

ON   specifies error recording is to be enabled for the specified device.

OFF   specifies error recording is to be disabled for the specified device.

Functional Details:

The number of errors recorded for the specified device is set to zero at system initialization.

Messages:

ERRC-ERR

indicates that error recording was not specified at sysgen.

NOPR-ERR

indicates that no parameters were specified.

NSUP-ERR

indicates the device does not support error recording.

PARM-ERR

indicates that a syntax error exists in the specified parameter.

3.31 EXAMINE COMMAND

The EXAMINE command is used to examine the contents of local or shared memory.

Format:

$$\text{EXAMINE address}_1 \left[\left\{ \begin{array}{c} ,n \\ /address_2 \\ ,1 \end{array} \right\} \right] \left[,\left\{ \begin{array}{c} fd \\ sys\ console \end{array} \right\} \right]$$

Parameters:

address   indicates the starting and ending addresses in memory whose contents are to be displayed in hexadecimal. All addresses specified are rounded down to halfword boundaries by the operating system.

n   is a decimal number specifying the number of halfwords to be displayed. If n is omitted, one halfword is displayed.

fd   is the file descriptor specifying the file or device to which the contents of memory are displayed; if omitted, the display is output to the system console.

Functional Details:

Specifying only address1 causes the contents of memory at that location to be displayed (as modified by any previous BIAS command.) Specifying address1 and address2 causes all data from the first to the second address to be displayed.

Example:

```
BI   B000                    examines  10  halfwords  starting at
EXA  100,10                  relative   address   100   (absolute
                             address B100).

EXA  100/200                 examines relative locations 100-200.
```

Messages:

ASGN-ERR

    indicates that an optional fd could not be assigned; e.g., fd already is assigned for exclusive use.

FD-ERR

    indicates an invalid fd.

FORM-ERR

    indicates a command syntax error.

IO-ERR

    indicates that an I/O error was detected on output device or file.

NOPR-ERR

    indicates that a required operand is missing.

PARM-ERR

    indicates an operand syntax error (an attempt to examine memory reserved for memory access controller (MAC) or marked off and possibly nonexistent).

3.32 FFILE COMMAND

The FFILE command forward spaces to the next filemark on magnetic tapes, cassettes, and direct access files.

Formats:

FFILE fd   used for magnetic tapes and cassettes only

FFILE fd [,lu]   used for disk devices only

Parameters:

fd   is the file descriptor of the device or file which is to be forward spaced one filemark.

lu   is the logical unit to which the file is assigned.

Functional Details:

For magnetic tapes and cassettes, only the parameter fd should be specified; for direct access files, lu optionally can be specified. The account number must be 0, if specified.

Before entering the format for disk devices, the task must be selected as the current task through the TASK command.

Examples:

FF MAG2:   causes the device MAG2: to forward space one filemark.

FF M300:AJM.OBJ,4   causes the file AJM.OBJ on volume M300 that is assigned to lu 4, to forward space one filemark.

Messages:

ASGN-ERR

        indicates that the file or device could not be assigned for
the reason noted in the TYPE field. See Appendix B for
possible entries in the TYPE field.

FD-ERR

        indicates that an invalid fd was encountered or a nonzero
account number was specified.

IO-ERR

        indicates that an I/O error or an illegal or unassigned lu
was encountered on the specified device or file.

LU-ERR

        indicates that the lu was not a legal decimal number or was
greater than maxlu for the task.

NOBC-ERR

        indicates that bulk file command support is not included in
the operating system.

NOPR-ERR

        indicates that no operand was specified.

TASK-ERR

        indicates that there was no currently selected task and an
FFILE command was entered with lu specified.

3.33 FRECORD COMMAND

The FRECORD command forward spaces one record on magnetic tapes, cassettes, and direct access files.

Formats:

FRECORD fd   used for magnetic tapes and cassettes only

FRECORD fd [,lu]   used for disk devices only

Parameters:

fd   is the file descriptor of the device or file to be forward spaced one record.

lu   is the logical unit to which the device or file is assigned.

Functional Details:

For magnetic tapes and cassettes, only the parameter fd should be specified; for direct access files, lu can optionally be specified. The account number must be 0, if specified.

Before entering the format for disk devices, the task must be selected as the current task through the TASK command.

Examples:

FR MAG2:   causes the device MAG2: to forward space one record.

FR M300:AJM.OBJ,4   causes file M300:AJM.OBJ on volume M300 that is assigned to lu 4 to forward space one record.

Messages:

ASGN-ERR

      indicates that the file or device could not be assigned for the reason noted in the TYPE field.

FD-ERR

      indicates that an invalid fd was encountered or a nonzero account number was specified.

IO-ERR

      indicates that an I/O error or an illegal or unassigned lu was encountered on the specified device or file.

LU-ERR

      indicates that the lu was not a legal decimal number or was greater than maxlu for the task.

NOBC-ERR

      indicates that bulk file command support is not included in the operating system.

NOPR-ERR

      indicates that no operand was specified.

TASK-ERR

      indicates that there was no currently selected task and a command requiring the specification of lu was entered.

3.34 INIT COMMAND

The INIT command enables the operator to initialize all data on
a contiguous file to 0.

Format:

$$\text{INIT fd} \quad \left[, \left\{ \begin{array}{c} \text{segsize increment} \\ 1 \end{array} \right\} \right]$$

Parameters:

fd   is the file descriptor of any unassigned,
                    unprotected, contiguous file. A file with a
                    nonzero account number can be initialized by
                    specifying the account number.

segsize   is a decimal number from 0 through 960kb |
increment   specifying the size of the buffer space used. |
                    The default is 1kb. |

Functional Details:

INIT is implemented with a CSS procedure that loads and starts
the file manager support utility as a background (.BG) task.

Examples:

    INIT DATA.FIL   initializes the file DATA.FIL.

    INIT DATA2.FIL,50   initializes the file DATA2.FIL using
                               a 50kb buffer.

Messages:

ASGN-ERR

    indicates an error when an attempt was made to assign the
    file to be initialized or the task file for the file manager
    support utility. See the ASSIGN command description for
    error information.

FD-ERR

indicates that an invalid fd was specified.

fd IS NOT A CONTIGUOUS FILE

indicates that an attempt has been made to initialize a
noncontiguous file.

FORM-ERR

indicates that an invalid segment size increment was
specified.

LOAD-ERR

indicates an error when an attempt was made to load the file
manager support utility as a background (.BG) task. See the
LOAD command for error information on loading a background
task.

MNEM-ERR

indicates that the file INIT.CSS could not be found on the
system volume.

NODA-ERR

indicates that there is no direct access support in this
system.

SEQ-ERR

indicates that another CSS procedure or background task is
active. A second INIT command cannot be executed until the
first has completed.

xxxx ERROR ON fd SECTOR n

indicates that an I/O error occurred while attempting to
initialize sector n of file fd. xxxx is the type of error;
it may be unrecoverable I/O, recoverable I/O, or device
unavailable.

3.35 IRBUFFER COMMAND

The IRBUFFER command is used to create, display, and free internal reader buffers that are used by the internal reader facility (SVC 2, 14) of the operating system.

Format:

$$\text{IRBUFFER} \begin{Bmatrix} n \\ \text{display} \\ \text{free} \end{Bmatrix}$$

Parameters:

n   is a decimal number from 1 through 99 specifying the number of command buffers requested.

display   displays how many buffers are available and how many are currently in use (have executable commands in them).

free   specifies that existing buffers are to be deleted.

Functional Details:

If the operating system is not generated with internal reader support (IREADER), any attempted use of the IRBUFFER command results in a mnemonic error (MNEM-ERR).

Examples:

```
IRB 5
IRBUF -   5 BUFFER(S) PRESENT
IRBUF -   0 BUFFER(S) IN USE

IRB D
IRBUF -   5 BUFFER(S) PRESENT
IRBUF -   0 BUFFER(S) IN USE

IRB F
IRBUF -   5 BUFFER(S) FREED
```

| Messages:

| IRBUF-BUFFERS CAN'T GET SYSTEM SPACE

| indicates that the requested number of buffers specified by
| IRBUFFER n exceeds the amount of available system space. To
| avoid this condition, fewer buffers should be requested, or
| the size of system space should be increased. Specifying the
| FREE option or entering IRBUF 0 releases all buffers.

| IRBUF-ILLEGAL PARAMETER

| indicates that the number of buffers requested by IRBUFFER n
| exceeds the maximum 99 buffers.

| can also indicate that IRBUFFER was entered with no
| parameter.

| IRBUF-n BUFFER(S) IN USE

| indicates that fewer buffers were requested than were
| currently in the queue. n represents the number of buffers
| in use. If this condition exists, the operating system will
| not reallocate the size of the buffer pool; the currently
| allocated buffer pool remains in effect.

| IRBUF-nn BUFFER(S) FREED

| indicates that the IRBUFFER command was entered with the FREE
| option, and all buffers are inactive.

| IRBUF-nn BUFFER(S) IN USE

| indicates that the IRBUFFER command was entered with the FREE
| option, and all buffers are active.

3.36 LOAD COMMAND

The LOAD command loads background tasks, foreground tasks, system
tasks, task common segments, and library segments into memory.

Format:

$$
\text{LOAD}\begin{cases}
\text{fd}\,[\text{, segsize increment}] \\
\text{taskid}\,[\text{, }[\text{fd}]\ [\text{, segsize increment}]] \\
\text{sysid, fd}\,[\text{, segsize increment}] \\
\text{.BG, fd}\,[\text{, segsize increment}] \\
\text{.TCM, fd} \\
\text{.LIB, fd} \\
\text{.SEG, fd}
\end{cases}
$$

Parameters:

fd

is the filename used as the taskid or the fd
of the established task, library, or
preinitialized task common segment to be
loaded into memory. If this parameter is
omitted, the default is taskid.TSK for
foreground tasks. This parameter is required
for system tasks, background tasks, task
common, and library segments. However, if the
extension of the fd is omitted, the default
extension is fd.SEG for a task common segment,
fd.SEG for a library segment, and fd.TSK for
foreground and background tasks.

segsize
increment

is a decimal number in kb specifying the
amount of get storage area in the task's
impure memory segment. If specified, this
value overrides the OPTION WORK= values used
when the task was linked.

taskid

specifies the name of the task after it is
loaded into the foreground segment in memory.

sysid | specifies the taskid of a system task. The taskid for the Spooler is .SPL; the taskid for the multi-terminal monitor is .MTM.

.BG | specifies that a background task is to be loaded into memory with intertask communication control and capabilities disabled.

.TCM | specifies that a preinitialized task common segment is to be loaded into memory. It is loaded as a .SEG segment.

.LIB | specifies that a library segment is to be loaded into memory. It is loaded as a .SEG segment.

.SEG | specifies that a task common or an RTL segment is to be loaded into memory. The default extension for fd is .SEG.

Functional Details:

A task must be prepared by processing the component programs, subroutines, and overlays with Link. Once established, the task can be loaded into memory. A task is loaded into the first memory segment large enough to accommodate it. A nonzero account number can be specified in the fd.

System tasks are extensions of the operating system. Currently existing system tasks are the .MTM, the Spooler (.SPL), and the SPL/32 (.SPLR).

Task common segments within OS/32 fall into two classes:

● those in local memory (below MTOP, as set by the sysgen MEMORY statement or the MEMORY operator command), and

● those in global memory (above MTOP).

Local task common segments are under operator control. The commands to add, delete, name, and set the size of local task common segments are described below. The number of local task common segments is limited only by the amount of memory available.

Global task common segments are established at sysgen time using the SYSGEN32 TCOM command, and are not controlled by the operator. The size and segment names are fixed at sysgen time. For more information on generating a system containing global task common segments, refer to the OS/32 System Generation/32 (SYSGEN/32) Reference Manual.

The library segment name specified at Link time is the name by
which the library is known to the system.

When a task is loaded into memory, the impure segment size
defaults to the size established at Link time, adjusted to a page
boundary. Certain utility and applications tasks require various
storage area sizes, depending on the particular execution. The
common assembly language (CAL/32) task, for example, requires a
variable work area in which to build a symbol table. The actual
size required for CAL/32 is a function of the size and number of
symbols in the program being assembled.

The segsize increment field of the LOAD command gives the user
the capability to vary the task's segment size at LOAD time, and
to override the amount of memory requested by the OPTION WORK=
command entered at Link time. If a task is established with a
minimum amount of get storage area, the memory available can be
increased or decreased with the LOAD command.

If a task requiring a TCOM or run-time library (RTL) is loaded,
the command processor attempts to load the required segments if
they are not already in memory. In order to accomplish this the
fd must be identical to the segment name (including the
extension). This is done by specifying an fd only in the BUILD
command of Link and letting segment name default to the fd. The
command processor searches the user volume/user account, then the
system volume/system account to locate the shared segment. This
allows MTM users some flexibility. When loading from the system
console, the shared segment must be on the system volume in the
system account.

Examples:

        L ABC,PTRP: loads a task from the paper tape
                                    reader punch device (PTRP:).

        L .BG,VOL:CAL   loads a task from file VOL:CAL.TSK
                                    into the background segment.

        L T1   loads a task from file T1.TSK on the
                                    default system volume into the
                                    foreground. Associates the name T1
                                    with the task.

        L T1,T1   loads a task from the default system
                                    volume from file named T1 into the
                                    foreground. Associates the name T1
                                    with the task.

        L OSCOPY,X:PSCOPY,1.5   loads a task from file X:PSCOPY.TSK
                                    into the foreground. Associates the
                                    name OSCOPY with the task.
                                    Specifies an expand area of 1.5kb.

L .BG,CAL,50.5	loads a task from the default system volume on file CAL.TSK into the background. Specifies an expand area of 50.5kb.
L .TCM,TASKCOM or L .SEG,TASKCOM.TCM	loads a preinitialized task common segment from file TASKCOM.TCM into memory.
L .SPL,SPOOLER	loads the Spooler system task from the default system volume on file SPOOLER.TSK into memory.
L .LIB,RTL or L .SEG,RTL.RTL	loads a reentrant library segment from file RTL.RTL into memory.

Messages:

FD-ERR

indicates an fd syntax error.

FORM-ERR

indicates a command syntax error.

LOAD-ERR TYPE=

indicates that load failed for reason noted in TYPE field.

NOPR-ERR

indicates that an operand is missing.

PARM-ERR

indicates an operand syntax error.

LOAD error TYPE field responses:

IO

indicates an I/O error on input file.

LIB

indicates an invalid format on LIB.

MAP

shared segment table (SST) is full or file was not found when an automatic attempt was made to load a sharable segment.

NAME

indicates that fd was not found or cannot be assigned.

NOFP

indicates that task requires floating point facilities not supported by sysgen.

OPT

indicates that task was not established as a system task.

PRES

indicates task is already present.

ROIO

indicates an I/O error in writing ROLL file.

RVOL

indicates an allocation or assignment error on roll file.

SEG

indicates that required library or task common is not present.

SPAC

indicates that there is not enough system space to complete load (for TCB).

SYS

    indicates that there was not enough system space for segment
    descriptor entry.

TKID

    indicates invalid taskid syntax.

USE

    indicates an attempt to load a task common over one of the
    same name that is presently in use.

3.37 MARK COMMAND

The MARK command takes a device offline or brings online a device
that was previously offline. For directory devices, the size   of
a volume's secondary directory and the expansion size can also be
specified with the MARK command.

Format:

$$
\text{MARK dev:,} \begin{Bmatrix} \text{PARAMETER } [\text{,RECL=recl}] \ [\text{,SIZE=size}] \\ \text{OFF} \\ \text{ON} \ \begin{bmatrix} \text{,} \begin{Bmatrix} \text{PROTECTED} \\ \text{SYSTEM } [/\text{max acct}] \\ \text{RESTRICTED } [/\text{max acct}] = \text{actno} \end{Bmatrix} \end{bmatrix} \ \begin{bmatrix} \text{,CDIRECTORY } \begin{bmatrix} = \begin{Bmatrix} \text{bsize} \\ 80 \end{Bmatrix} \\ \text{ALL} \end{bmatrix} \begin{bmatrix} / \begin{Bmatrix} \text{exp} \\ 100 \end{Bmatrix} \end{bmatrix} \end{bmatrix} \end{Bmatrix}
$$

Parameters:

dev:	is the device mnemonic.
OFF	marks a device offline.
ON	marks a device online.
PARAMETER	indicates that the record length and form size of a device are being modified.
RECL=recl	recl is a decimal number from 1 through 255 specifying the new record length for the output device. If this parameter is omitted, the default is 80.
SIZE=size	size is a decimal number from 1 through 88 specifying the form length for the output device. If this parameter is omitted, the default is 66.
PROTECTED	marks a device write-protected.
SYSTEM	inhibits write access to the device by a task running under a nonzero account under MTM.

> Account 255 is the "owner" of a
> disk marked on as SYSTEM.

RESTRICTED=   actno is the account number of the owner of
               the restricted device. The device is marked
               on with restricted access indicating that only
               the owner has access to it unless otherwise
               specified via the RVOLUME command under MTM.

max acct   is the maximum account number which is allowed
               any access to a SYSTEM or RESTRICTED device.
               If specified, the value will be rounded up to
               the next multiple of 256 (minus 1). If
               omitted, no account above 255 will be allowed
               to access (read or write) the device.

CDIRECTORY   creates a secondary directory in memory.

bsize   is the number of secondary directory entries
               which will be contained in the memory-resident
               buffer for the secondary directory. The
               minimum buffer size is 20 file names (256
               bytes); the maximum size is limited to the
               size of system space. If bsize is not
               specified, the default value 80 is used. If
               bsize is not a multiple of 20, it is rounded
               up to the next higher multiple of 20.

exp   is the expansion size for the secondary
               directory file on disk. This is the number of
               files that can be allocated before the
               secondary directory file overflows. If the
               expansion size is omitted, the default is 100.

ALL   specifies that the disk is to be marked online
               with a secondary directory large enough to
               maintain all directory slots in memory (i.e.,
               bsize=SLOTS, exp=0, and PAGES=1; see
               explanation for SLOTS and PAGES). If enough
               memory space is available, it is suggested
               that the ALL parameter be used.

Functional details for marking a device on or off:

The command is rejected if it is directed to:

● the system console,

● the NULL device,

- a direct access device containing files that are currently assigned, or

- a device that is currently assigned.

After marking on a direct access device, the volume name associated with it is output to the console device in the format device mnemonic: volume name, (e.g., D300: M300). While a device is offline, it cannot be assigned to any u-task. E-tasks are permitted to assign offline devices.

If the device being marked on or off is a direct access device, the fd used in the command is not the volume identifier, but the actual device mnemonic. For example, to mark off a disk named DSCl, which currently contains a volume named SYS1, the operator enters:

        MA DSCl:,OF

This action removes the volume SYS1 from the system. The disk can now be removed or changed if DSCl is a removable cartridge disk. To make the new volume known to the system, the operator enters:

        MA DSCl:,ON

This causes the volume descriptor of the pack on DSCl to be read. The volume id associated with DSCl is output to the system console.

                              NOTE

        Removable   cartridges   should   not be
        dismounted   from   the   system   without
        marking them off.

If the optional parameter PROTECTED is specified in a MARK ON command, the device is marked as write-protected. All assignments for access privileges other than SRO and SRW are rejected with a privilege error. SRW is changed to SRO. A WFILE (write filemark) command to any file on the device is also rejected. The PROTECT option can be used for any device regardless of the state of the hardware write-protect feature. The PROTECT option must be specified for hardware protected disks.

If a direct access device is dismounted without being marked offline, it can be marked online only in the write protected mode. The FASTCHEK utility must be run before the volume can be marked online without the PROTECT option. This ensures the integrity of files on a pack inadvertently dismounted. However, if the direct access device dismounted was originally marked online with the PROTECT option, the disk can be remounted and marked online without the PROTECT option.

If an I/O error occurs on a disk while the disk is being marked off, the following message is displayed on the system console:

        I/O ERROR MARKING OFF DISK ; PLEASE CHECK

The disk is marked off and can be marked online only in the write protected mode. The FASTCHEK utility must be run before the volume can be marked online without the PROTECT option.

When marking a direct access device on with a secondary directory, additional information is displayed.

Example:

        MA DSC1:, ON,,CD=120/0
        DSC1: OS32 CDIR   220 FILES   240 SLOTS   2 PAGES   1.85 K

DSC1:	is the device name of the direct access device.
OS32	is the volume name of the disk.
CDIR	specifies that the disk was marked on with a secondary directory.
FILES	is the number of files on disk.
SLOTS	is the number of directory slots on disk (i.e., the total of the number of active files and the number of free directory slots currently on the disk).
PAGES	is the number of pages of secondary directory pages in SYSTEM.DIR. In the above example, the disk contains 240 directory slots and was marked on to maintain 120 directory slots in memory. Each page of SYSTEM.DIR will contain 120 directory slots requiring two pages to contain all 240 directory slots. Only one page of SYSTEM.DIR is maintained in the in-memory buffer at a time.

K is the number of kbytes of system space used for the secondary directory.

For direct access devices, file access time and consequently system performance can be optimized through the correct use of device directories. Three types of directories are available for use on direct access devices:

- Default primary directory (disk resident)

- Fast access primary directory (disk resident)

- Secondary directory (memory resident and disk resident)

A default primary directory is a noncontiguous directory comprised of a series of 1-sector blocks. Each block can contain directory information for up to five files. These directory blocks are allocated when needed as the number of files on a disk increases. When a directory block is filled, (five files allocated on the disk) another directory block is allocated on the next free sector encountered on the disk. This type of directory provides slow file access time due to the noncontiguous format of the directory and provides the least-optimal file access times. This type of directory is used by default if a fast access primary directory or secondary directory is not specified.

A fast access primary directory is allocated at disk initialization time via the BLOCKS= option of the disk initializer utility (see the OS/32 System Support Utilities Reference Manual), or by FASTCHEK (see OS/32 Fastchek Reference Manual) via the BLOCKS= or DIRECTORY= options. When specified, the BLOCKS= or DIRECTORY= option preallocates a contiguous area on the disk to hold the primary directory. The capacity of the directory is based on the number of blocks specified. One block can hold directory information for five files. The contiguous format of this type of directory results in faster access times in comparison to the noncontiguous format of the default primary directory.

A secondary directory can be created when a disk is brought online via the MARK ON command. A secondary directory provides optimal access times, and consequently, improved system performance by placing a directory of files on the disk directly into an in-memory buffer. The in-memory directory is searched, when necessary, for file directory information. When a disk is marked online with the secondary directory option specified, a contiguous file called SYSTEM.DIR, large enough to hold all entries in the primary directory plus any expansion size, is automatically allocated on the disk. This file constitutes the secondary directory. All or part of the contents of SYSTEM.DIR are then transferred into an in-memory buffer, depending upon the capacity of the buffer.

The capacity of the buffer is specified via the bsize entry in this command:

        MARK disk:,ON,,CD

If the buffer capacity is sufficient to contain the total number of file entries in SYSTEM.DIR, the entire contents of the secondary directory are placed into the memory buffer (i.e., one page). If the buffer capacity is not sufficient to contain the total number of file entries in the secondary directory, the contents of the directory are paged into the memory buffer during file access operations. Optimal system performance and access times are obtained if the memory buffer is large enough to contain the entire secondary directory.

<center>NOTE</center>

When specifying the capacity of a secondary directory memory buffer for disks containing large numbers of files, the amount of memory used by the buffer can be prohibitive. Anticipate the size of the buffer using the following formula:

$$\text{memory space used (in bytes)} = \frac{bsize*256}{20}$$

Therefore, a disk with 3,000 files requires 38.4kb of memory to accommodate the entire secondary directory:

$$\frac{3000*256}{20} = 38400 \text{ bytes} = 38.4kb$$

The following examples are presented to illustrate the relationship of the fast primary directory, the secondary directory, the secondary directory expansion size, and the size of the in-memory secondary directory buffer.

Example 1:

A disk volume contains 2,400 files and was initialized with a fast access primary directory of BLOCKS=550 or DIRECTORY=2750. Thus, the fast access primary directory contains space for 2,750 files (1 block equals 5 files). To establish an optimal performance secondary directory, use the following command.

MARK disk:,ON,,CD=2750/0

This command allocates a secondary directory and gives the
in-memory buffer the capacity to hold 2,750 file entries. No
expansion capability is included. The memory buffer will use
35.2kb of system space.

Example 2:

A disk volume contains 2,100 files and was initialized with a
fast access primary directory of BLOCKS=400. The fast access
primary directory has overflowed the preallocated size area by
100 files. Assume that 500 additional files will be allocated on
the disk before it is marked off. To establish an optimal
performance secondary directory, the following command is used:

        MARK disk:,ON,,CD=2600/500

This command allocates a secondary directory on the disk and
gives the in-memory buffer the capacity to hold 2,600 files. The
expansion factor of 500 allows another 500 files to be allocated
on the disk. The memory buffer will use 33.3kb of system space.

If many files are created on a disk that is marked on with a
small expansion factor, it is possible that the SYSTEM.DIR file
may become full. Further attempts to allocate files result in
secondary directory overflow. In each case, a message:

        CDIR FULL-fd

is displayed on the system console, where fd is the device
mnemonic of the disk. This message is displayed only on the
system console. When secondary directory overflow occurs, the
system starts using both the secondary and the primary
directories when searching for a file. In this case, file
accessing time may be greater. To overcome this problem, the
disk can be marked off and then marked on again with a desired
expansion factor at a convenient time. A disk can also be marked
on protected and include a secondary directory, provided a good
SYSTEM.DIR file is present on that disk.

Functional details for marking on a disk as restricted:

When a disk is marked on restricted, the owner of the disk can allow other users to access it and can assign access privileges via the RVOLUME command under MTM. The Spooler and Backup are affected by this as shown in Table 3-6. For information on how MTM is affected, see the OS/32 Multi-Terminal Monitor (MTM) Reference Manual.

| A disk can be marked ON with RESTRICTED=0. This will prevent any
| access by MTM users. Note, however, that once this is done,
| access cannot be granted to any account.

TABLE 3-6   EFFECTS OF RESTRICTED DISKS ON SPOOLER AND BACKUP

TASK	READ/WRITE ACCESS TO ACCOUNT 0	READ ONLY ACCESS TO ACCOUNT 0	NO ACCESS TO ACCOUNT 0
.SPL and .SPLR	All access is valid.	Files can be printed but not deleted. If a delete is attempted, the system operator receives an error message.	Files cannot be accessed.
BACKUP	All access is valid.	Verify only option is valid. If backup is attempted, an assign error occurs.	If access is attempted, an assign error occurs.

Functional details for modifying parameters of an output device:

Specifying the PARAMETERS option indicates that the record length and form size of an ASCII output device are being modified. The
| device for which these parameters are being modified cannot be assigned. Modification for a pseudo device being used by the Spooler must be made by canceling the Spooler, specifying the new parameters, and restarting the Spooler.

When modifying the parameters, either parameter can be specified.
| If both are omitted, the current record length and page size are
| displayed.

Examples:

MA DSC1:,ON,P,CD	verifies the primary directory with the existing secondary directory. 80 files will be maintained in the memory resident buffer for the secondary directory. Expansion factor, if specified, is ignored.
MA DSC1:,ON,P,CD=200	200 files will be maintained in the memory resident buffer for the secondary directory.
MA DSC1:,ON,R=4	marks a disk on restricted, allowing only the owner of account 4 to have access to it. Accounts above 255 are allowed no access to the disk.
MA DSC2:,ON,SYS/1023,CD=ALL	disk DSC2: is marked online as a system disk. Accounts 1 through 254 and 256 through 1023 have read only access; accounts zero and 255 have read/write access. The secondary directory will be created with an in-memory buffer large enough to hold the entire directory.
MA PRT:,PA,R=65,S=56	changes the record length and form size of device PRT: to 65 characters per line and 56 lines per page.

Messages:

BPAC-ERR

indicates that a direct access volume I/O error was encountered.

DEV-ERR

indicates that an attempt was made to mark on or off a nonexistent device, or a pseudo device, or the NULL device; or an attempt was made to modify the attributes of a nonexistent device or bulk storage device.

DUPL-ERR

indicates that a duplicate device or volume name exists.

FORM-ERR

indicates a command syntax error.

| **NOFF-ERR**

indicates that an attempt was made to mark on a direct access device not previously marked offline before dismounting.

PARM-ERR

indicates an operand syntax error.

READ-ERR

indicates that the device is not hardware enabled, a hardware error exists, or a disk pack is bad.

STAT-ERR

indicates that the fd was already assigned or has files assigned.

WRIT-ERR

indicates that a device is hardware write-protected or disk pack is bad.

If an attempt to mark a disk on with a secondary directory option fails, the disk is marked on without the secondary directory, and one of these messages is generated:

FETCH ATTR ERR

indicates that fetch attributes failed on secondary directory when marking protected with secondary directory.

NO SYSTEM SPACE

indicates that not enough system space was available for creating a secondary directory block.

PRI-DIR READ ERR

indicates that an I/O error was encountered when reading the primary directory.

SEC-DIR ALLO ERR

indicates failure to allocate sufficient space on the disk for creating new secondary directory, or that the disk is write protected (hardware feature).

SEC-DIR ASGN ERR

indicates that assignment to secondary directory failed during mark on protected with secondary directory.

SEC-DIR DELE ERR

indicates that a DELETE command to old secondary directory failed because old file was not properly closed. In this case, a disk integrity check should be run on that disk.

SEC-DIR NOT PRESENT

indicates that secondary directory does not exist on disk and mark on protected with secondary directory was attempted.

SEC-DIR READ ERR

indicates that an I/O error was encountered when reading the secondary directory during marking on in protected mode.

SEC-DIR VERIFY ERR

indicates that secondary directory failed to verify with the primary directory when marking protected with secondary directory.

SEC-DIR WRIT ERR

indicates that an I/O error was encountered when establishing the secondary directory on the disk.

```
------------
|   MEMORY   |
------------
```

3.38 MEMORY COMMAND

The MEMORY command specifies the memory area (in kb) to be designated as local memory, or performs run-time memory diagnostics on a specified memory area.

Format:

$$
\text{MEMORY} \begin{Bmatrix} \underline{\text{OFF}}\text{,address } [\text{,size}] \\ \text{ON,address } [\text{,size}] \\ \underline{\text{TEST}}\text{,address } [\text{,size}] \\ n \end{Bmatrix}
$$

Parameters:

OFF | marks OFF a block making that area unavailable. This parameter also can be used to mark off the following:

- an area from which a memory module is to be removed, and

- an area when an online test is to be run.

ON | marks ON a memory block making that area available.

TEST | tests a marked off block for bad memory areas.

n | is a positive decimal number from 1 through 16384 specifying the size of local memory in kb. The default value set at sysgen is overridden by n.

address | is a hexadecimal number specifying the starting address of the memory area to be marked off, on, or tested.

size | is a decimal number that is a multiple of the processor's block size specifying the number of kilobytes in memory to be marked off, tested, or marked on. If this parameter is omitted, the default is one block.

Functional Details:

On Models 7/32, 8/32, and 3220 systems the block size is 256 bytes and the maximum local memory size is 1024kb; on Models 3210, 3230, 3240, 3250, and 3200MPS Systems, the block size is 2048 bytes and the maximum local memory size is 16,384, except the 3210, which is limited to 4096kb.

Memory diagnostics consist of initial and run-time memory testing. Before loading the operating system into memory, the loader storage unit (LSU) tests the area of memory into which the operating system will be loaded. The LSU then loads the operating system into memory and transfers control to the operating system. The operating system tests memory from the end of the operating system to the end of local memory. The operating system does not perform memory tests on global memory. If bad or unavailable memory exists, it is marked off, and the marked off area is noted on the memory map when the DISPLAY MAP command is entered.

Before memory testing begins, the following message is displayed:

MEMORY TEST IN PROGRESS

Memory testing is then performed on physical memory in block increments. Any bad or missing memory is marked off. Adjacent marked off blocks are concatenated and labeled OFF MEMORY on the memory map. At the end of initial memory testing, a count of bad blocks is displayed:

UNAVAILABLE MEMORY BLOCKS = nnnn

The decimal count of bad blocks that have been marked off is nnnn. Entering the DISPLAY MAP command displays the location and size of each block. At the end of run-time testing, the following message is displayed:

NO. OF BAD MEMORY BLOCKS = xxxx

The decimal count of bad blocks is xxxx. A count of zero indicates no bad memory was found in the tested area. If the count is nonzero, a memory map is displayed. When memory testing is over, the following message is displayed:

END OF MEMORY TEST

The following approximate times for testing memory apply to a 7/32 processor. The figures can be reduced by 50 percent for other processors:

- The LSU requires 10 seconds to test memory and load the standard operating system into 128kb of memory.

- Initial memory testing requires 60 seconds to test 1Mb of memory.

- Use of the test parameter requires 10 milliseconds to test 1kb of memory.

The following four sections further describe functions of memory diagnostics.

1. Marking Off Memory

If a memory parity error occurs during execution of a u-task, the affected task is paused with its memory map address displayed when the DISPLAY MAP command is entered.

Use the OFF parameter to mark off the bad memory area before checkpointing and canceling the task. Failure to mark off the bad memory will cause data in the task memory to be destroyed. After the memory is marked off and the task canceled, the TEST parameter tests the marked off area.

If a memory parity error or memory malfunction occurs on a Perkin-Elmer 3200 Series processor and the operating system is sysgened with memory test support (MEMCHECK), the operating system automatically marks off the affected block, removing it from task space without operator intervention. When the OFF parameter is entered, the required memory area must be located in task or system space. If it is in system space, above system space, or between system space, an address or size message is displayed and memory is not marked off. If the block is in system space, that area must be free, otherwise a message is displayed.

Areas are marked off in memory starting with an address that is a multiple of the processor block size. A block is the smallest area that can be marked off. Rounding off is done by dropping the least-significant address bit. It is legal to mark off an area already marked off or marked off pending. No errors are reported.

If the area to be marked off is in task space containing free and allocated space, the affected free space is marked off. The affected allocated space is flagged as marked off pending, and it remains marked off pending until released by a terminated task. Then that area is automatically marked off.

2. Marking On Memory

Before memory is marked on, a memory diagnostics test is performed.

The affected area must have been marked off or marked off pending and must be located within a single block. It is legal to mark on only a portion of the marked off area. It is also legal to mark on an area which is already marked on. No errors are reported.

Marked on memory is added to the free task space or free system space. If the requested memory was marked off pending, the pending condition is removed.

Memory areas are marked on starting with an address that is a multiple of the memory block size. A block is the smallest area that can be marked on.

3. Testing Memory

The MEMORY command which specifies a test parameter tests a memory area after it has been marked off. The test runs at the command processor's priority, causing lower priority tasks to be locked out until the test has been completed.

These restrictions apply to this command:

● The area to be tested must be marked off (not marked off pending), and must exist within task or system space (not cross task or system space boundaries).

● It is legal to test a portion of the marked off memory area.

● The minimum memory area to be tested is a block.

● Memory areas are tested starting with an address that is a multiple of the block size.

● All data in the tested memory area is destroyed.

4. Specifying the Size of Local Memory

The MEMORY command with the specified n parameter can be entered only when no tasks are in the system, before any disks are marked on with a secondary directory, and before setting a disk device as the log device.

Examples:

```
MEMORY ON,46600,2.25     marks on memory 46600-46EFF (2.25kb)
                         for Models 7/32, 8/32, and 3220
                         processors.    Marks    on    memory
                         46000-46FFF  (4kb)  for  the  Models
                         3210, 3230, 3240, 3250, and 3200MPS
                         processors.

MEMORY OFF,3167E         marks off an area 31600-316FF
                         (0.25kb) for the Models 7/32, 8/32,
                         and 3220 processors. Marks off an
                         area 31000-317FFF  (2kb)  for  3210,
                         3230,  3240,  3250,  and  3200MPS
                         processors.

MEMORY OFF,4663C,2.25    marks off an area 46600-46EFF
                         (2.25kb) for the Models 7/32, 8/32,
                         and 3220 processors. Marks off an
                         area 46000-46FFF (4kb) for the 3210,
                         3230,  3240,  3250,  and  3200MPS
                         processors.

MEMORY TEST,46600,2.25   Tests an area 46600-46EFF (2.25kb)
                         for the Models 7/32, 8/32 and 3220
                         processors.    Tests    an    area
                         46000-46FFF  (4kb)  for  the  Models
                         3210, 3230, 3240, 3230, 3240, 3250
                         and 3200MPS processors.
```

The following sequence of commands illustrates marking off and marking on memory for the Models 7/32, 8/32, and 3220 processors:

```
*DISPLAY MEMORY
        NAME        TYPE     START    SIZE     SEG    SYS    STAT    PRI
TASK MEMORY                  1D500    550.75
OFF MEMORY                   1D500    1.00                          OFF
OFF MEMORY                   1DB00    0.50                          OFF
SYSTEM SPACE                 A7000    100.00
*MEMORY ON,1D500
NO. OF BAD MEMORY BLOCKS = 0
*DISPLAY MEMORY
        NAME        TYPE     START    SIZE     SEG    SYS    STAT    PRI
TASK MEMORY                  1D500    550.75
OFF MEMORY                   1D600    0.75                          OFF
OFF MEMORY                   1DB00    0.50                          OFF
SYSTEM SPACE                 A7000    100.00
*MEMORY ON,1D700
NO. OF BAD MEMORY BLOCKS = 0
*DISPLAY MEMORY
        NAME        TYPE     START    SIZE     SEG    SYS    STAT    PRI
TASK MEMORY                  1D500    550.75
OFF MEMORY                   1D600    0.25                          OFF
OFF MEMORY                   1D800    0.25                          OFF
OFF MEMORY                   1DB00    0.50                          OFF
SYSTEM SPACE                 A7000    100.00
```

To merge blocks marked off and marked off pending, follow the sequence of commands below. This example pertains to the Models 7/32, 8/32, and 3220 processors.

```
*DISPLAY MEMORY
        NAME            TYPE    START    SIZE    SEG    SYS    STAT    PRI
TASK MEMORY                     1D500    550.75
SYSTEM SPACE                    A7000    100.00
*TCOM TC1,0.25
*TCOM TC2,0.25
*MEMORY OFF,1D500,0.5
*DISPLAY MEMORY
        NAME            TYPE    START    SIZE    SEG    SYS    STAT    PRI
TASK MEMORY                     1D500    550.75
TC1             .SEG            1D500      0.25
OFF MEMORY                      1D500      0.25                 PEND
TC2             .SEG            1D600      0.25
OFF MEMORY                      1D600      0.25                 PEND
SYSTEM SPACE                    A7000    100.00
*
REM .SEG,TC2
*DISPLAY MEMORY
        NAME
                        TYPE    START    SIZE    SEG    SYS    STAT    PRI
TASK MEMORY                     1D500    550.75
TC1             .SEG            1D500      0.25
OFF MEMORY                      1D500      0.25                 PEND
OFF MEMORY                      1D600      0.25                 OFF
SYSTEM SPACE                    A7000    100.00
*
REM .SEG,TC1
*DISPLAY MEMORY
        NAME            TYPE    START    SIZE    SEG    SYS    STAT    PRI
TASK MEMORY                     1D500    550.75
OFF MEMORY                      1D500      0.50                 OFF
SYSTEM SPACE                    A7000    100.00
*
```

This sequence of commands illustrates memory testing for the Models 7/32, 8/32, and 3220 processors.

```
*DISPLAY MEMORY
        NAME            TYPE    START    SIZE    SEG    SYS    STAT    PRI
TASK MEMORY                     1D500    550.75
OFF MEMORY                      1D500      4.00                 OFF
SYSTEM SPACE                    A7000    100.00
*MEMORY TEST,1D500,4
NO. OF BAD MEMORY BLOCKS = 3
                ADDR    SIZE
BAD   BLK       1D500    0.75
GOOD  BLK       1D800    1
BAD   BLK       1DC00    1
GOOD  BLK       1E000    1
BAD   BLK       1E400    0.25
```

Messages:

FORM-ERR

    indicates a command syntax error.

MEM-ERR

    indicates that n is less than UBOT, greater than physical memory, or greater than the starting address of global task common. The system contains no memory available for system space.

MEM-ERR TYPE=ADDRESS

    indicates that address is outside task or system space.

MEM-ERR TYPE=FIND

    indicates that requested system space is not free or requested memory is located in another segment.

MEM-ERR TYPE=NOMD

    indicates that the MEMORY command was entered, but the system does not support memory diagnostics.

MEM-ERR TYPE=SIZE POS=XXX

    indicates that calculated end address is between task and system space or is outside system space. XXX is the requested value.

MEM-ERR TYPE=SPAC

    indicates that insufficient system space exists to mark off the specified memory area; memory is not marked off.

NOPR-ERR

    indicates that an operand is missing.

PARM-ERR

    indicates that parameter is invalid.

SEQ-ERR

    indicates that the system is not quiescent.

```
------------
|  MODIFY   |
------------
```

3.39 MODIFY COMMAND

The MODIFY command is used to modify the contents of local or shared memory.

Format:

$$
\underline{\text{MO}}\text{DIFY address} \left[, \left\{ \begin{matrix} \text{data}_1 \\ 0 \end{matrix} \right\} \right] \left[, \text{data}_2, \ldots, \text{data}_n \right]
$$

Parameters:

address   is the address at which the contents of memory are to be modified.

data   is a data field consisting of 0- to 4-hexadecimal digits that represent a halfword to be written into memory starting at the location specified by address. Any string of data less than four characters is right-justified and left-zero filled. If the comma but no data is entered, 0 is entered into one halfword. If data is omitted, a message is displayed.

Functional Details:

This command causes the contents of the halfword location specified by address (modified by any previous BIAS command) to be replaced with data. The modify address must be aligned on a halfword boundary.

Example:

BI 0   modifies four halfwords at location
MOD 12F0,4,0,4,0   12F0 to contain 0004 0000 0004 0000.

FORM-ERR

   indicates a command syntax error.

PARM-ERR

   indicates an operand error occurred because an address was
   not aligned on a halfword boundary, an address specified was
   not in memory or was reserved for MAC, or a specified address
   was within the marked off section of memory.

```
  ------------
| OPTIONS    |
  ------------
```

3.40 OPTIONS COMMAND

The OPTIONS command is used to specify or change certain options
of the currently selected task.   An OPTIONS command can be
entered if the referenced task is dormant or paused.   For users
of the Model 3200MPS System, two new options have been added to
the command format. They are: LPU [=n] and NLPU.

Format:

$$\underline{O}PTIONS \quad \begin{bmatrix} \begin{Bmatrix} \underline{AF}CONT \\ \underline{AFPAUSE} \end{Bmatrix} \end{bmatrix} \begin{bmatrix} , \begin{Bmatrix} \underline{RES}IDENT \\ \underline{NONRESIDENT} \end{Bmatrix} \end{bmatrix} \begin{bmatrix} , \begin{Bmatrix} \underline{SVCC}ONTINUE \\ \underline{SVCPAUSE} \end{Bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} , \begin{Bmatrix} \underline{NROLL} \\ \underline{ROLL} \end{Bmatrix} \end{bmatrix} \begin{bmatrix} , \begin{Bmatrix} LPU \; [=n] \\ NLPU \end{Bmatrix} \end{bmatrix}$$

Parameters:

AFCONT	specifies that if the arithmetic fault (AF) trap enable bit is set, a trap is taken. If the bit is not set, the task continues after arithmetic fault occurs and the message is sent to the log device.
AFPAUSE	specifies that the task is to pause after any arithmetic fault.
RESIDENT	specifies that the task is memory resident.
NONRESIDENT	specifies that the task is to be removed from memory at end of task.
SVCPAUSE	specifies that SVC 6 is treated as an illegal SVC (applies to .BG only). If an SVC 6 is executed within a background task, the task is paused.
SVCCONTINUE	specifies that SVC 6 is treated as NOP (applies to .BG only). If an SVC 6 is executed within a background task, the task is continued.

ROLL   specifies that the task can be rolled.

NROLL   specifies that the task cannot be rolled.

LPU=n   sets the task as an LPU directed task (a task |
                        directed to execute on a specific logical |
                        processing unit). The specific LPU can be |
                        specified by n. |

NLPU   sets the task as a CPU directed task (a task |
                        which must execute on the CPU). If the APU |
                        ONLY option was set at Link time, NLPU is |
                        invalid and an OPT-ERR occurs. |

Functional Details:

Unless otherwise specified with the Link OPTIONS command, when a
task is loaded, the default options are: AFPAUSE, NONRESIDENT,
SVCPAUSE, ROLL, and, for Model 3200MPS Systems, NLPU.   If |
accounting is enabled in the system, ACCOUNT also becomes a
default option for a task. An OPTIONS ROLL command is invalid
unless the directed task is linked as rollable.

If conflicting options are specified, the latest option entered
is accepted.   Thus, OPTIONS RESIDENT, NONRESIDENT specifies |
NONRESIDENT. |

The AFPAUSE and AFCONT options are normally set up at Link time,
but the console operator can modify them.

The sequence TASK taskid, OPT NONRESIDENT, and CANCEL taskid |
always causes the currently selected task to be removed from
memory. The sequence TASK taskid, OPT RESIDENT, and CANCEL
taskid always causes the currently selected task to enter the
dormant state.

The SVCPAUSE and SVCCONTINUE parameters apply only to a
background task.   They are ignored if they are specified for a
foreground task.

If an OPTIONS NROLL is directed to a task currently rolled out,
the option goes into effect after the task is rolled into memory.

If any parameter is invalid, previous valid parameters in the
same command are processed. In this case, the DISPLAY PARAMETERS
command can be used to verify the state of the task options.

For users of the Model 3200MPS System, the following information applies:

The LPU and NLPU parameters do not affect the task options word; instead they affect the LPU directed task status. The LPU parameter sets the LPU directed task status and assigns n (if specified) as the task's LPU number. LPU=0 assigns the CPU by system convention (illegal if APU ONLY task option set at Link time). The NLPU parameter resets the LPU directed task status (i.e., task is CPU directed). If the combination LPU=N, NLPU is entered, the task's LPU number is set to n and the task is CPU directed. This assigned LPU number has no effect until the task is changed to LPU directed.

Messages:

FORM-ERR

    indicates a command syntax error.

NROL-ERR

    indicates invalid option; task not rollable.

OPT-ERR

    indicates invalid option; roll option not specified at Link time; non-APU execution is prohibited at Link time.

PARM-ERR

    indicates an operand syntax error.

SEQ-ERR

    indicates that task is not dormant or paused.

TASK-ERR

    indicates that there is no currently selected task.

3.41 PAUSE COMMAND

The PAUSE command causes the currently selected task to pause.

Format:

    PAUSE

Functional Details:

Any I/O proceed, ongoing at the time the task is paused, is allowed to continue to completion. If the task is in any wait state at the time the PAUSE command is entered, all external wait conditions must be satisfied before the pause becomes effective. This command is rejected if the task is dormant or paused at the time the command is entered.

For users of the Model 3200MPS System, this command applies also | to tasks active or ready on an APU, in which case, the task is | removed from the APU's control. |

Messages:

FORM-ERR

     indicates a command syntax error.

SEQ-ERR

     indicates that task is paused or dormant.

TASK-ERR

     indicates that there is no currently selected task.

```
------------
|   REMOVE    |
------------
```

3.42 REMOVE COMMAND

The REMOVE command is used to remove from memory a shared segment
that is loaded by a LOAD command or established by a TCOM
command. The memory area it occupied is freed for system use.

Format:

REMOVE .SEG,segment name

Parameters:

.SEG   specifies that a shared segment is to be
                removed.

segment name   is the symbolic name of the shared segment.

Messages:

FORM-ERR

    indicates a command syntax error.

PARM-ERR

    indicates an operand syntax error.

REM-ERR

    indicates that an attempt to remove a segment failed for
    reasons noted in TYPE field. See Appendix B for possible
    entries in the TYPE field.

3.43 RENAME COMMAND

The RENAME command is used to change the name of an unassigned direct access file or device.

Format:

        RENAME oldfd,newfd

Parameters:

oldfd   is the current file descriptor of the file or device to be renamed.

newfd   is the new file descriptor to which the file or device is renamed.

Functional Details:

The volume id field of the new file descriptor (newfd) can be omitted for direct access files. If it is entered, the system ignores it. This command cannot be used to rename a direct access volume; the FASTCHEK utility must be used for this. | Attempts to rename the console device or the null device or to rename a real or a pseudo device to an existing device or volume result in an error. A file can only be renamed if its write and read protection keys are 0 (X'0000').

By specifying different account numbers in the oldfd and newfd, a file can be logically transferred from one user to another (the file is not copied). Files in account 255 cannot be transferred in this way.

Examples:

REN VOL:AJM.CUR,AJM.NEW   renames file AJM.CUR to AJM.NEW on
                          volume VOL.

REN MT01:,MT02:   renames device MT01 to MT02.

Messages:

ASGN-ERR TYPE=PRIV POS=REN

    indicates that fd is currently assigned, or the RENAME is directed to the system console.

ASGN-ERR TYPE=PROT POS=fd

    indicates that protection keys are nonzero.

FD-ERR

    indicates an invalid fd.

FORM-ERR

    indicates a command syntax error.

NOPR-ERR

    indicates that a required parameter is missing.

NULL-ERR

    indicates an attempt to rename the null device.

PARM-ERR

    indicates an operand syntax error.

RENM-ERR TYPE=BUFF POS=fd

    indicates that an error occurred when closing an lu for a RENAME, or the system space control blocks are corrupted.

RENM-ERR TYPE=NAME POS=REN

    indicates that a duplicate device or volume name exists.

RENM-ERR TYPE=NAME POS=fd

    indicates that a duplicate filename exists.

3.44 REPROTECT COMMAND

The REPROTECT command permits the operator to modify the protection keys of an unassigned direct access file or device.

Format:

    REPROTECT fd,new keys

Parameters:

fd	is the file descriptor of the file or device to be reprotected.
new keys	is a hexadecimal halfword whose most-significant byte signifies the new write keys and whose least-significant byte signifies the new read key.

Functional Details:

If a file with a nonzero account number is to be reprotected, the account number field must be specified in the fd. This facility is provided mainly to enable changing the keys to 0 so that a nonzero account number file can be renamed or deleted. Unconditionally protected files or devices can be conditionally reprotected or unprotected. See the ASSIGN command, Section 3.3.

Messages:

ASGN-ERR

    indicates that reprotect failed for the reason noted in TYPE
    field. See Appendix B for possible entries in the TYPE
    field.

FD-ERR

    indicates an invalid fd.

FORM-ERR

   indicates a command syntax error.

PARM-ERR

   indicates an operand syntax error.

PRIV-ERR

   indicates an attempt to reprotect a pseudo spool device.

REPR-ERR

   indicates that reprotect failed for the reason noted in TYPE
   field.   See Appendix B for possible entries in the TYPE
   field.

3.45 REWIND AND RW COMMANDS

The REWIND and RW commands rewind magnetic tapes, cassettes, and direct access files. Either command is entered from the system console.

Formats:

REWIND fd   used for magnetic tapes and
  or   cassettes only
RW fd

REWIND fd [, lu]   used for disk devices only
  or
RW fd [, lu]

Parameters:

fd   is the file descriptor of the device or file
                to be rewound.

lu   is the logical unit assigned to the device or
                file.

Functional Details:

For magnetic tapes and cassettes, only the parameter fd should be specified; for direct access files, lu optionally can be specified. The account number must be 0, if specified.

Before entering the format for disk devices, the task must be selected as the current task through the TASK command.

Examples:

REW MAG1:   causes the tape on device MAG1: to
                         be rewound.

REW M300:AJM.OBJ,4   causes file AJM.OBJ, as assigned to
                         lu 4 on volume M300, to be rewound.

Messages:

ASGN-ERR

    indicates that the file or device could not be assigned for the reason noted in the TYPE field. See Appendix B for possible entries in the TYPE field.

FD-ERR

    indicates an invalid fd or the file has a nonzero account number.

FORM-ERR

    indicates a command syntax error.

IO-ERR

    indicates that an I/O error or an illegal or unassigned lu was encountered on the specified device or file.

LU-ERR

    indicates that an invalid fd was encountered or a nonzero account number was specified

NOPR-ERR

    indicates that no operand was specified.

PARM-ERR

    indicates an operand syntax error.

TASK-ERR

    indicates that there was no currently selected task and a REWIND command with lu specified was entered.

ASGN error TYPE field responses:

NAME

    indicates that fd does not exist on specified volume.

PRIV

    indicates that file or device is already assigned.

VOL

    indicates that volume or device does not exist.

```
------------
|  RVOLUME   |
------------
```

3.46 RVOLUME COMMAND

The RVOLUME command allows the system operator to display to the
system console the accounts that have access to a privately owned
restricted disk.

Format:

    RVOLUME voln

Parameter:

    voln   is the volume name of the restricted disk.

Functional Details:

When an operator displays the accounts having access to a
privately owned disk, the accounts are displayed at the system
console along with their access privileges.

Example:

    RVOL M300
    OWNER = 20
    0/RO   20/RW   22/RO   36/RO   80/RW   88/RW

Messages:

PARM-ERR

    indicates a parameter is missing or invalid.

PRIV-ERR

    indicates that the specified volume is not a restricted disk.

VOLN-ERR

    indicates that the volume is not online or the volume name is
    invalid.

3.47 SEND COMMAND

The SEND command is used to send a message to the currently selected task.

Format:

    SEND message[;]

Parameter:

message   is a variable length string of 1- to 64-alphanumeric characters.

Functional Details:

The message is passed to the selected task in the same manner as an SVC 6 send message. Following standard SVC 6 procedures, the message consists of an 8-byte taskid identifying the system manager, followed by the operator-supplied character string. The message data passed to the selected task begins with the first nonblank character following SEND and ends with a carriage return or semicolon (;) as a line terminator. A message cannot be sent to a task currently rolled out.

The receiving task must have intertask message traps enabled in its TSW and must have established a message buffer area. Since background tasks cannot have intertask message traps enabled, a message cannot be sent to a task in the background. Refer to the OS/32 Supervisor Call (SVC) Reference Manual for more information on the SVC 6 send message function.

Example:

    TASK ANITA
    SEND CLOSE LU2,ASSIGN LU3

produces the following message for the task ANITA:

    .CMDP CLOSE LU2, ASSIGN LU3

Messages:

ARGS-ERR

    indicates that a message exceeded 64 characters.

NOPR-ERR

    indicates that no message was provided. The first nonblank character following the SEND command was a carriage return.

SEQ-ERR

    indicates that task is paused, not yet started, or not capable of receiving a message.

SVC6-ERR

    indicates that an SVC 6 error was returned specifying that the task could not receive a message trap.

TASK-ERR

    indicates that there was no currently selected task.

3.48 SET LOG COMMAND

The SET LOG command is used to specify the system log device. The system log device receives all system console I/O. This includes:

● all command lines entered from the console or from the CSS,

● all responses to these commands (other than prompts), and

● all messages logged by tasks.

Format:

$$\text{SET LOG [fd] [,COPY]} \left[, \left\{ \begin{array}{c} n \\ 15 \end{array} \right\} \right]$$

Parameters:

fd   is the file descriptor of the log device.

COPY   specifies that a copy of all system console
                I/O is to be sent to the system console |
                device, as well as to the log device. |

n   is a decimal number from 0 to 65,535
                specifying the number of lines after which the
                system log file is to be checkpointed. If
                this parameter is omitted, the default is 15
                lines.

Functional Details:

If COPY is specified, the system console continues to receive all system console I/O and a copy is sent to the log device. If COPY is not specified, the system console receives nothing.

Checkpointing is only meaningful for indexed files on disk. If n is specified as 0, no checkpointing occurs. If n is omitted, the default is 15 lines. This ensures that any lines not checkpointed will not be visually lost and can be manually recorded by the operator, if necessary.

The log device can be shared with u-task output. It can be changed at any time by another SET LOG command. If no parameters are specified, logging is terminated. Logging is automatically terminated under these conditions:

- I/O error on the log device

- System initialization

- Power restoration

When logging is terminated, the system console device again receives all output.

SET LOG is primarily used for:

- providing a historical record of system operation, often on magnetic tape or direct access file; and

- allowing system output displays, log messages, etc., to proceed on a high-speed device rather than on the system console.

Example:

SET LOG PR:,COPY   sets the log device to a printer (PR:), and sends a copy of all system console I/O to the system console device as well as to the log device.

Messages:

ASGN-ERR

  indicates that a log device or file could not be assigned; that is, device offline or assigned for exclusive use to task.

FD-ERR

indicates an invalid fd.

IO-ERR

indicates that an I/O error occurred on output device or file.

PARM-ERR

indicates an operand or command syntax error.

```
------------
|   SET     |
| PRIORITY  |
------------
```

3.49 SET PRIORITY COMMAND

The SET PRIORITY command is used to modify the priority of the currently selected task.

Format:

    SET PRIORITY n

Parameter:

    n   is a decimal number ranging from a high priority of 10 through a low priority of 249 specifying the new priority for the currently selected task.

Functional Details:

The priority of the currently selected task is set to n, subject to the following restriction. If the task is a foreground task, its priority cannot exceed the maximum priority set when the task was built. To increase the maximum priority, reestablish the task. The maximum priority a background task can attain is set at sysgen. To increase this priority, resysgen the system. The default priority for all tasks is 128.

For Model 3200MPS Systems, the priority assigned to a task affects scheduling on the CPU only.

Example:

    TA ABC   sets the priority of task ABC to
    SET PRI 150   150.

Messages:

FORM-ERR

    indicates a command syntax error.

NOPR-ERR

    indicates that required parameter n was omitted.

PARM-ERR

    indicates a parameter syntax error.

PRTY-ERR

    indicates that new priority is greater than maximum priority
    specified at Link time.

TASK-ERR

    indicates that there was no currently selected task.

3.50 SET SLICE COMMAND

The SET SLICE command is used to invoke the time-slice scheduling option. For the Model 3200MPS System, the option applies to the CPU only.

Format:

SET SLICE n

Parameter:

n   is zero or a decimal number from 20 to 65,536
                           specifying the time-slice allocated to each
                           task.

Functional Details:

If n is 0, time-slice scheduling is disabled; otherwise, n
represents the maximum time in milliseconds that any one task can
remain active if another task of equal priority is ready. The
time-slice option is initially disabled.

Two types of scheduling algorithms are available. Tasks can be
scheduled   in   strict   priority   order   or   time-sliced within
priority.   In the former case, if two tasks of equal priority are
started, a task remains active until it relinquishes control of
the processor. Assign priorities carefully so that tasks that do
not frequently relinquish control of the processor do not
inadvertently lock out other tasks.   A task can relinquish
control in one of the following ways:

● Task is paused by the console operator.

● Task is cancelled by the operator or another task.

● A higher priority task becomes ready because of some external
event.

● Task executes an SVC that places it in wait, pause, or dormant
state.

Rather than scheduling on a strict priority basis, tasks can be time-sliced within priority. This option allows the user to ensure that tasks of equal priority receive equal shares of processor time. When a task becomes ready, it is queued on a round-robin basis behind all ready tasks of equal priority. For Model 3200MPS Systems, the task can elect to relinquish the remainder of the time-slice by use of the Reschedule (RSCH) instruction.

Example:

remain active to 20 milliseconds if another task of equal priority is ready.

Messages:

FORM-ERR

indicates a command syntax error.

NOPR-ERR

indicates that operand n was missing.

PARM-ERR

indicates that operand n is not 0 or a decimal number within the required bounds.

3.51 SET SYS COMMAND

The SET SYS command is used to vary the size of system space. System space is used to hold system control blocks and buffers (TCBs, FCBs, etc.).

Format:

SET SYS n

Parameter:

n   is a decimal number from 0.25 through 16,384 specified in increments of .25kb. The value n is rounded up to the next page boundary.

Functional Details:

This command is rejected if insufficient memory is available. It is also rejected if a decrease in system space size is requested and the current usage exceeds the new size. Otherwise, it is accepted whenever contiguous memory is available to accommodate the request. The SET SYS command cannot cause a rollout of any task.

The OS/32 data structures and file control block (FCB) requirements for each are:

Contiguous files	280 bytes
Indexed files	528 + blocking factor x 512 + index block factor x 256
Nonbuffered files	528 + index block factor x 256
Timer queue entries	24 bytes
Task control blocks	560 + 4 x maximum number of logical units + 12 x number of accounting I/O classes

Example:

SET SYS 75.25   adjusts the amount of system space available to 75.25kb for a Model 7/32, 8/32, or 3220 system. On a 3210, 3230, or 3240 system the amount of system space is adjusted to 76kb.

Messages:

FORM-ERR

    indicates a command syntax error.

MEM-ERR POS=XXX

    indicates insufficient memory. XXX is the requested value.

NOPR-ERR

    indicates required operand is missing.

PARM-ERR

    indicates an operand or command syntax error, or that n is not a decimal number specified in increments of .25kb.

```
------------
|  SET TIME  |
------------
```

3.52 SET TIME COMMAND

The SET TIME command sets the current date and time of day.

Format:

$$\text{SET TIME} \left\{ \begin{array}{c} [\text{mm/dd/yy}], \text{hh:nn:ss} \\ \text{OFF} \\ 0 \end{array} \right\}$$

Parameters:

mm	is a decimal number from 01 through 12 specifying the month.
dd	is a decimal number from 01 through 31 specifying the day.
yy	is a decimal number from 00 through 99 specifying the year.
hh	is a decimal number from 00 through 23 specifying the hour.
nn	is a decimal number from 00 through 59 specifying minutes.
ss	is a decimal number from 00 through 59 specifying seconds.
OFF	turns off the clock for system debugging.
0	turns off the clock for system debugging.

Functional Details:

The SET TIME command must be entered when the system is first loaded and after any power failure occurs. It may be entered whenever the system clock is incorrect. The day, month, and year are automatically updated by the system (even during leap years).

At system initialization, following the display of the operating system id, this message is output to the system console:

ENTER DATE AND TIME

If a command other than SET TIME is entered, the prompt ENTER DATE AND TIME is repeated until a SET TIME command is entered. If a SET TIME command is entered while incomplete time intervals exist, the tasks that initiated the incomplete intervals are affected in these ways:

- Seconds from midnight - The time and day are updated; however, this has no effect on any time of day interval even if the date entered differs from the date previously entered. The time difference is used to adjust all seconds from midnight intervals.

- Milliseconds from now - Elapsed time intervals are unaffected by a change in the time by a SET TIME.

For example, if the current date is 11/22/80 and the current time is 11:50 AM, and there are three intervals outstanding:

1. Time of day interval is set to complete on 11/22/80 at 2:00 PM.

2. Time of day interval is set to complete on 11/23/80 at 8:00 AM.

3. Elapsed time interval is set to complete on 11/22/80 at 1:00 PM.

If SET TIME 11/21/80, 10:50:00 is entered, the time intervals are as follows:

1. Time of day interval is set to complete on 11/21/80 at 2:00 PM.

2. Time of day interval is set to complete on 11/22/80 at 8:00 AM.

3. Elapsed time interval is set to complete on 11/21/80 at 12 noon.

When the system is restarted after a power failure, the clock is automatically turned on, and the date and time at the time of power failure is used. The operator should correct the time after the power failure.

During system debugging, it is often desirable to eliminate interrupts. The SET TIME command provides the system debugging feature. This feature is performed by entering:

$$\text{SET TIME} \left\{ \begin{array}{c} 0 \\ \text{OFF} \end{array} \right\}$$

System debugging mode may be selected at any time. When this mode is selected, the clock is turned off, clock interrupts are inhibited, the display panel is cleared, and the previous date and time setting (if previously set) is cleared. The operator may exit from this mode at any time by entering SET TIME with the desired date and time specified.

Example:

    SET TIME 2/24/80,03:05:00

Alternatively, by sysgen option, the date parameter can be entered in the format:

    dd/mm/yy

The date parameter is optional only if it has been previously set. Therefore, at system start-up, both the date and time must be specified.

Messages:

FORM-ERR

    indicates a command syntax error.

NOPR-ERR

    indicates that an operand is missing.

PARM-ERR

    indicates an operand syntax error.

3.53 SPOOLFILE COMMAND

The SPOOLFILE command allows a user to allocate a spool file on behalf of a specified pseudo device and assign that file to a specified lu of the currently selected task. This command makes all spooling options available at a terminal or CSS level. This command is not processed by SPL/32 as a spooler command. It is handled by the operating system and MTM.

The SPOOLFILE command cannot be used with the OS/32 spooler.

Format:

$$\text{SPOOLFILE lu&lu1,pseud dev,FORM=formname} \left[,\begin{Bmatrix} VFC \\ IMAGE \end{Bmatrix}\right]$$

$$\left[,\begin{Bmatrix} NOIMAGE \\ NOVFC \end{Bmatrix}\right] \left[,\begin{Bmatrix} CHECKPOINT \\ NOCHECKPOINT \end{Bmatrix}\right] [,COPIES=n] \left[,\begin{Bmatrix} HOLD \\ RELEASE \end{Bmatrix}\right]$$

$$[,BLOCK= blocksize/indexsize] \left[,\begin{Bmatrix} DELETE \\ NODELETE \end{Bmatrix}\right] [,PRIORITY=p]$$

Parameters:

lu is a decimal number specifying the logical unit to which the pseudo device is to be assigned.

lu1 indicates that lu is to be assigned to the same spool file as lu 1. lu 1 must be the first lu assigned to the spool file.

pseud dev is the 1- to 4-character name of a pseudo device. The first character must be alphabetic; the remaining alphanumeric.

FORM= is a desired preprinted form name that can be specified here. If the form specified was not previously enabled using a FORM command, an error message is sent to the monitoring control or subcontrol task and the request is processed using the default standard form name, STD.

VFC	specifies the use of vertical forms control for the assigned lu. When VFC is used, the first character of each record is interpreted as a VFC character. If VFC is not included, there is no VFC for the device assigned to the specified lu.
NOVFC	turns the vertical forms control option off for the assigned lu. This is the default option.
COPIES=	identifies the number of copies to be output. It must be between 1 and 255 or an error message is sent.
HOLD	causes the specified file to remain on the spool queue until a RELEASE request is issued.
RELEASE	enables a spool file for output when the lu is closed.
BLOCK	specifies the index and/or data block size.
blocksize	is a decimal number specifying the physical block size in 256-byte sectors, to be used for buffering and debuffering operations involving the file. The default size is 1 or the value entered using the BLOCK command. If this value exceeds the maximum block size established at sysgen time, an error will be printed when attempting to allocate the file.
indexsize	is a decimal number specifying the index block size in 256-byte sectors. The default size is 1 or the value entered using the BLOCK command. Index size cannot exceed the maximum index block size established at sysgen time or an error will occur when attempting to allocate the file.
DELETE	the file is deleted after output. This is the default option.
NODELETE	the file is not deleted after output.
PRIORITY=p	p is the desired print priority. If this option is not specified, the print priority becomes the same as the priority of the task the spool file assign is on behalf of.

Functional Details:

The SPOOLFILE command can be used to make an assignment to a pseudo device from the terminal or CSS level. If two conflicting parameters are entered in a single SPOOLFILE command, such as DELETE and NODELETE, the second parameter is executed and an error message is generated. The SPOOLFILE command is primarily for users operating in an MTM environment.

Example:

    SPOOLFILE 4,pdl:,VFC,DELETE

                              This example causes a spool file to be allocated for pseudo device pdl: and assigns that file to lu 4 of the current task. VFC has been specified for the specified lu and the DELETE option has been selected, which means the file will be deleted after output.

```
-------------
|   START   |
-------------
```

3.54 START COMMAND

The START command is used to initiate task execution. The currently selected task is started only if it is dormant; otherwise, the command is rejected.

Format:

$$\text{START} \left[\left\{ \begin{array}{c} \text{address} \\ \text{transfer address} \end{array} \right\} \right] \left[\text{,parameter}_1 \text{,} \ldots \text{,parameter}_n \right]$$

Parameters:

address
: specifies the address at which task execution is to begin. For u-tasks, this is not a physical address, but is an address within the task's own logical address space. For e-tasks, it is a physical address. If address is omitted, the currently selected task is started at the transfer address specified when the task was established.

parameter
: specifies optional parameters to be passed to the task for its own decoding and processing. Parameters must be separated by a comma. All characters between the first comma and the command terminator (carriage return or semicolon) are moved to memory beginning at UTOP. The characters are terminated in memory by a carriage return. If this operand is omitted, a carriage return is stored at UTOP. If there is not enough memory between UTOP and CTOP to pass all the characters, the call is rejected with an ARGS-ERR.

Examples:

    ST 138   starts the currently selected task at X'138'.

    ST 100,NOSEG,SCRAT   starts the currently selected task at X'100' and passes NOSEG,SCRAT to the task.

    ST ,1000,ABC   starts the currently selected task at transfer address specified when the task was built, and passes 1000,ABC to the task.

Messages:

FORM-ERR

indicates a command syntax error.

PARM-ERR

indicates an operand syntax error.

SEQ-ERR

indicates that task is active.

SVC6-ERR TYPE=ARGS

indicates insufficient memory between UTOP and CTOP to pass all parameters.

SVC6-ERR TYPE=DORM

indicates a START was entered but task is in time delay wait.

TASK-ERR

indicates that there was no currently selected task.

```
------------
|   TASK    |
------------
```

3.55 TASK COMMAND

The TASK command is used to specify the current foreground, background, or system task.

Format:

    TASK [taskid]

Parameter:

    taskid   is the name of a foreground task, background
                        task (.BG), or a system task (.MTM or .SPL).
                        If taskid is not specified, the taskid of the
                        currently selected task is displayed on the
                        console device.

Functional Details:

Task-related commands operate on the currently selected task. All task-related commands are affected by the TASK command. Also affected are the BFILE, BRECORD, FFILE, FRECORD, REWIND, RW, and WFILE commands. Some CSS commands are affected by TASK as described in Chapter 5.

NOTE

When CSS is started, the current value of TASK is associated with the CSS file as the currently selected CSS task. If a CSS file executes a TASK command, it affects only that CSS file's commands and does not change the value of TASK associated with the console.

If the currently selected task is deleted from the system, or if no TASK command has been entered, task-related commands (other than TASK) are rejected with a TASK-ERR.

Example:

T ABC	Sets current task to ABC.
CL 2,3,4	Closes logical units 2, 3, and 4 of task ABC.
AS 2,CARD:	Assigns lu 2 of task ABC to device CARD.
T XYZ	Sets current task to XYZ.
OPT RES	Makes task XYZ resident.
CAN XYZ	Cancels task XYZ.
ST 100	Starts task XYZ at relative address 100.
T .BG	Sets current task to background.
PAUSE	Pauses the background task.

Messages:

PARM-ERR

    indicates an operand syntax error.

TASK-ERR

    indicates that there was no currently selected task.

```
------------
|   TCOM   |
------------
```

3.56 TCOM COMMAND

The TCOM command is used to allocate an area of memory for a local task common segment to be referenced by u-tasks.

Format:

    TCOM segment name,segment size

Parameters:

   segment name   is the symbolic name of the common block used by programs in referencing the common segment.

   segment size   is the size of the memory area to be allocated. Size is a positive decimal number in kb, specified in multiples of 256-byte increments for the Models 7/32, 8/32, and 3220 processors; 2048-byte increments for the Models 3210, 3230, 3240, 3250, and 3200MPS processors, with the maximum value depending on the available task space.

Functional Details:

The segment is resident until removed via the REMOVE command.

Examples:

   TCOM COM1,2.5   allocates a 2.5kb task common segment COM1.

   TCOM COMM2,20   allocates a 20kb task common segment COMM2.

Messages:

FORM-ERR

    indicates a command syntax error.

PARM-ERR

    indicates an operand syntax error.

TCOM-ERR

    indicates that creation of task common segment failed for
    reason noted in TYPE field.

TCOM error TYPE field responses:

MEM

    indicates that there is no vacant area of sufficient size.

NAME

    indicates that task common of same name already exists.

SYS

    indicates that there is not enough system space for required
    segment descriptor entry (SDE).

3.57 TEMPFILE COMMAND

The TEMPFILE command causes a temporary file to be allocated and
assigned to one of the logical units of the currently selected
task. A temporary file exists only for the duration of the
assignment. When a temporary file is closed, it is also deleted.

Format:

$$
\text{TEMPFILE lu,}
\begin{cases}
\text{CONTIGUOUS,fsize} \\[4pt]
\text{EC}\left[/\left[\left\{{\text{bsize} \atop 64}\right\}\right]\right]\left[/\left[\left\{{\text{isize} \atop 3}\right\}\right]\right] \\[4pt]
\text{INDEX}\left[,\left[\left\{{\text{lrecl} \atop 126}\right\}\right]\right]\left[/\left[\left\{{\text{bsize} \atop 1}\right\}\right]\right]\left[/\left[\left\{{\text{isize} \atop 1}\right\}\right]\right] \\[4pt]
\text{NB}\left[,\left[\left\{{\text{lrecl} \atop 126}\right\}\right]\right]\left[/\left[\left\{{\text{bsize} \atop 64}\right\}\right]\right]\left[/\left[\left\{{\text{isize} \atop 3}\right\}\right]\right]
\end{cases}
$$
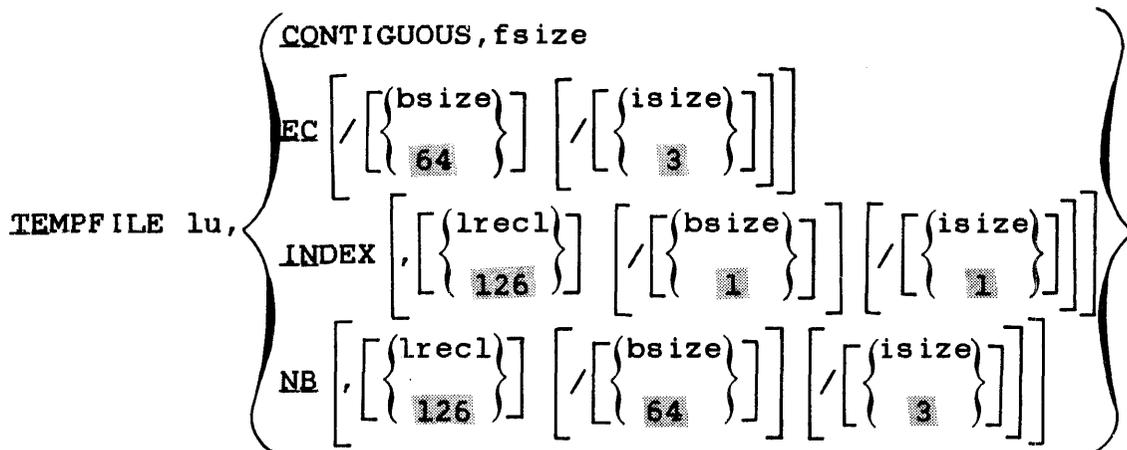
Parameters:

lu	is a decimal number specifying the logical unit of the currently selected task to which a temporary file is assigned.
CONTIGUOUS	specifies that the file type to be allocated is contiguous.
fsize	is a decimal number specifying the total allocation size in 256-byte sectors. This size can be any value up to the number of contiguous free sectors existing on the specified volume at the time the command is entered.
EC	specifies that the file type to be allocated is extendable contiguous.

bsize   is a decimal number specifying the physical
                         block size to be used for buffering operations
                         with indexed files, and for the data blocks
                         used for indexed, nonbuffered indexed, and
                         extendable contiguous files. When INDEX, EC,
                         or NB is specified, bsize represents the block
                         size in sectors of the physical data blocks
                         containing the file. For INDEX files, this |
                         parameter cannot exceed the maximum block size |
                         established by the sysgen procedure. For EC
                         and NB files, this parameter may be any value
                         from 1 to 255 inclusive. If bsize is omitted,
                         the default value for INDEX files is 256 bytes
                         (one sector). For EC and NB files, the
                         default is 64 sectors.

isize   is a decimal number specifying the index block
                         size. For INDEX files, the default value is
                         one sector (256 bytes). For EC and NB files,
                         the default value is three sectors (768
                         bytes). The index block size cannot exceed
                         the maximum diskblock size established by the
                         sysgen procedure. Neither bsize nor isize may
                         exceed 255.

INDEX   specifies the file type to be allocated is
                         buffered indexed.

lrecl   is a decimal number specifying the logical
                         record length of an indexed or nonbuffered |
                         indexed file, or a data communications device. |
                         It cannot exceed 65,535 bytes. Its default is |
                         126 bytes. |

NB   specifies that the file type to be allocated |
                         is nonbuffered indexed. |

Functional Details:

A temporary file is allocated on the default TEMP volume. Refer
to Section 3.57 for information on the TEMP volume.

To assign an indexed file, sufficient room must exist in system
space for three buffers (two data buffers and one index buffer), |
each of the stated size. Therefore, if bsize or isize is very
large, the file cannot be assigned in some memory-bound
situations. At sysgen, a maximum block size parameter is
established in the system, and bsize and isize cannot exceed this
constant.

To assign an EC or NB file, sufficient room must exist in system |
space to contain only a single index block of the stated size. |
The data blocks for EC and NB are not buffered in system space |
and, thus, are not constrained to the sysgened block size. |

| Examples: |

| TE 2,CO,64 | allocates on the default temporary volume a contiguous file with total length of 64 sectors (16kb) and assigns it to lu 2 of the currently selected task. |

| TE 14,IN,126 | allocates on the default temporary volume an indexed file with logical record length of 126 bytes. The buffer size and index block size default to one sector. The file is assigned to lu 14 of the currently selected task. |

| TE 4,EC | allocates on the default temporary volume an extendable contiguous file with the default data block size of 64 sectors and the default index block size of 3 sectors. The file is assigned to lu 4 of the currently selected task. |

| TE 12,NB,240/250 | allocates on the default temporary volume a temporary nonbuffered indexed file with a logical record length of 240 bytes, a data block size of 250 sectors, and the default index block size of 3 sectors. The file is assigned to lu 12 of the currently selected task. |

Messages:

ALLO-ERR

  indicates that allocation failed for reason denoted by TYPE field.

ASGN-ERR

  indicates that the assign failed for reason noted by the TYPE field. See Appendix B for possible entries in the TYPE field.

FORM-ERR

  indicates a command syntax error.

LU-ERR

   indicates an invalid lu number or lu assigned.

NODA-ERR

   indicates that direct access support is not included in this
   system.

NOPR-ERR

   indicates a required operand is missing.

PARM-ERR

   indicates an operand syntax error.

SEQ-ERR

   indicates that currently selected task is not dormant.

SPAC-ERR

   indicates that task exceeds established maximum system space
   usage.

TASK-ERR

   indicates that there is no currently selected task.

```
  -------------
|   VOLUME    |
  -------------
```

3.58 VOLUME COMMAND

The VOLUME command is used to set or change the name of the system, task, roll, spool, or temporary volume. Alternatively, it is used to interrogate the system for the current names associated with these volumes.

Format:

$$\underline{V}OLUME \left[voln \left[\Bigg/ \left\{ \begin{array}{l} \underline{S}YSTEM \\ \underline{T}ASK \\ \underline{T}EMPORARY \\ \underline{S}PL \\ \underline{R}OLL \end{array} \right\} \right] \right]$$

current voln names
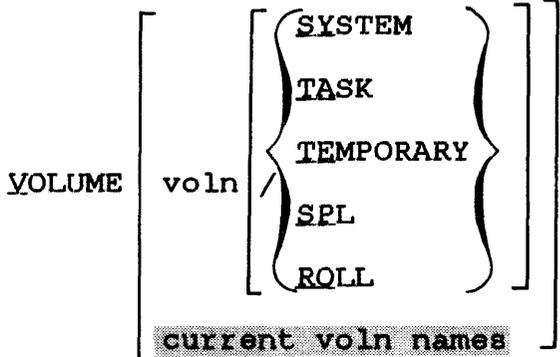
Parameters:

voln   is a 4-character volume identifier. If all
                parameters are omitted, all current system,
                task, roll, spool, and temporary volume names
                are displayed.

SYSTEM   specifies the system volume.

TASK   specifies the task volume of the currently
                selected task.

TEMPORARY   specifies the temporary volume.

SPL   specifies the spool volume.

ROLL   specifies the roll volume.

Functional Details:

Any commands that do not explicitly specify a volume name use the system volume as a default. No test is made to ensure that the volume is actually online at the time the command is entered. If voln is not specified, the names of the current default volumes are output to the console device.

Examples:

VOL ABCD/SPL	establishes volume ABCD as a spool volume.
V	displays all current volume names to
SYSTEMV=MT4	the console.
TASKVOL=MT4	
TEMPVOL=FIXD	
SPOOLV =FIXD	(will be displayed if spool support included at sysgen)
ROLLVOL=FIXD	(will be displayed if roll support included at sysgen)

Messages:

NODA-ERR

indicates that no direct access support was sysgened.

PARM-ERR

indicates an operand syntax error.

TASK-ERR

indicates that there is no currently selected task.

```
------------
|   WFILE    |
------------
```

3.59 WFILE COMMAND

The WFILE command writes a filemark on magnetic tapes, cassettes, and direct access files.

Formats:

    WFILE fd   used for magnetic tapes and cassettes only

    WFILE fd [, lu]   used for disk devices only

Parameters:

    fd   is the file descriptor of the file or device to which a filemark is to be written.

    lu   is the logical unit to which the device or file is assigned.

Functional Details:

For magnetic tapes and cassettes, only the parameter fd should be specified; for direct access files, lu optionally can be specified. The account number must be 0, if specified.

Before entering the format for disk devices, the task must be selected as the current task through the TASK command.

Examples:

    WF MAG1:   causes a filemark to be written on the magnetic tape on MAG1:.

    WF M300:AJM.OBJ,4   causes a filemark to be written on file AJM.OBJ, which is assigned to lu 4 on volume M300.

Messages:

ASGN-ERR

      indicates that the file or device could not be assigned for the reason noted in the TYPE field. See Appendix B for possible entries in the TYPE field.

FD-ERR

      indicates an invalid fd.

FORM-ERR

      indicates that a command syntax error was encountered.

IO-ERR

      indicates that an I/O error or an illegal or unassigned lu was encountered on the specified device or file.

LU-ERR

      indicates that an invalid lu was encountered.

NOPR-ERR

      indicates that no operand was specified.

PARM-ERR

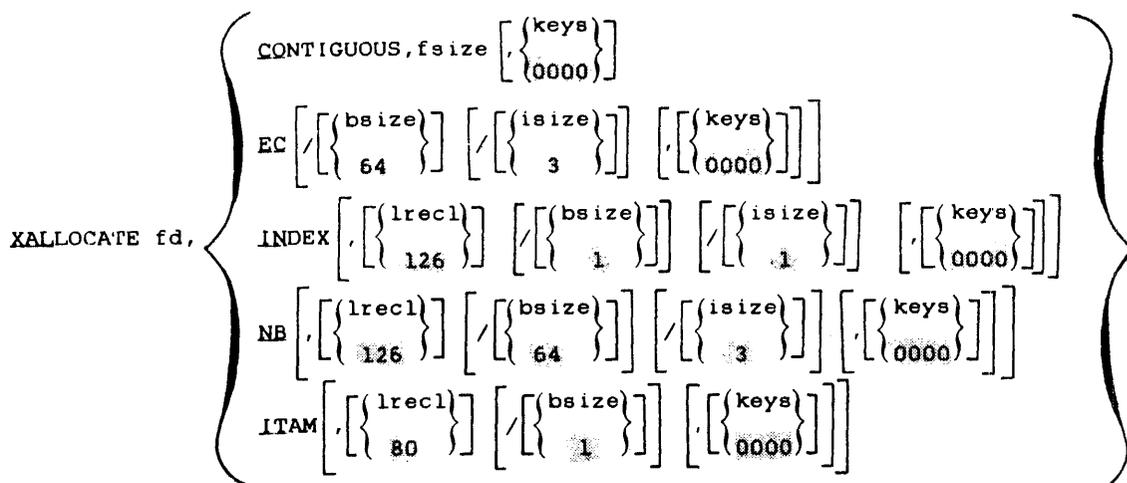   indicates that an operand syntax error was encountered.

TASK-ERR

      indicates that there was no currently selected task and a WFILE command with lu specified was entered.

3.60 XALLOCATE COMMAND

The XALLOCATE command deletes an existing file and allocates a
file with the same name.

Format:

$$
\text{XALLOCATE fd,}
\left\{
\begin{array}{l}
\underline{\text{CONT}}\text{IGUOUS,fsize}\left[,\left\{\begin{array}{l}\text{keys}\\0000\end{array}\right\}\right] \\[2ex]
\underline{\text{E}}\text{C}\left[\left[/\left\{\begin{array}{l}\text{bsize}\\64\end{array}\right\}\right]\left[/\left\{\begin{array}{l}\text{isize}\\3\end{array}\right\}\right]\left[,\left\{\begin{array}{l}\text{keys}\\0000\end{array}\right\}\right]\right] \\[2ex]
\underline{\text{IN}}\text{DEX}\left[,\left\{\begin{array}{l}\text{lrecl}\\126\end{array}\right\}\right]\left[/\left\{\begin{array}{l}\text{bsize}\\1\end{array}\right\}\right]\left[/\left\{\begin{array}{l}\text{isize}\\1\end{array}\right\}\right]\left[,\left\{\begin{array}{l}\text{keys}\\0000\end{array}\right\}\right] \\[2ex]
\underline{\text{N}}\text{B}\left[,\left\{\begin{array}{l}\text{lrecl}\\126\end{array}\right\}\right]\left[/\left\{\begin{array}{l}\text{bsize}\\64\end{array}\right\}\right]\left[/\left\{\begin{array}{l}\text{isize}\\3\end{array}\right\}\right]\left[,\left\{\begin{array}{l}\text{keys}\\0000\end{array}\right\}\right] \\[2ex]
\underline{\text{IT}}\text{AM}\left[,\left\{\begin{array}{l}\text{lrecl}\\80\end{array}\right\}\right]\left[/\left\{\begin{array}{l}\text{bsize}\\1\end{array}\right\}\right]\left[,\left\{\begin{array}{l}\text{keys}\\0000\end{array}\right\}\right]
\end{array}
\right\}
$$

Parameters:

fd
: is the file to be deleted and the name of the new file to be allocated.

CONTIGUOUS
: specifies that the file type to be allocated is contiguous.

fsize
: is a decimal number indicating file size, which is required for contiguous files. It specifies the total allocation size in 256-byte sectors. This size may be any value up to the number of contiguous free sectors existing on the specified volume at the time the command is entered.

keys   specifies the write and read protection keys
        for the file. These keys are in the form of
        a hexadecimal halfword, the most-significant
        byte of which signifies the write key and the
        least-significant byte, the read key. If this
        parameter is omitted, both keys default to 0.

EC   specifies that the file type to be allocated
        is extendable contiguous.

bsize   is a decimal number specifying the physical
        block size to be used for buffering and
        debuffering operations on indexed files or
        data communications devices, and for the data
        blocks used for indexed, nonbuffered indexed,
        and extendable contiguous files. When INDEX,
        EC, or NB is specified, bsize represents the
        block size in sectors of the physical data
        blocks containing the file. When ITAM is
        specified, bsize represents the buffer size in
        bytes. For INDEX files and ITAM buffers, this
        parameter cannot exceed the maximum block size
        established by the sysgen procedure. For EC
        and NB files this parameter may be any value
        between 1 and 255 inclusive. If bsize is
        omitted, the default value for INDEX files and
        ITAM buffers is 256 bytes (one sector). For
        EC and NB files, the default is 64 sectors.

isize   is a decimal number specifying the index block
        size. For INDEX files, the default value is
        one sector (256 bytes). For EC and NB files,
        the default value is three sectors (768
        bytes). The index block size cannot exceed
        the maximum disk block size established by the
        sysgen procedure. Neither bsize nor isize may
        exceed 255.

INDEX   specifies that the file type to be allocated
        is buffered indexed.

lrecl   is a decimal number specifying the logical
        record length on an indexed or nonbuffered
        indexed file, or a data communications device.
        It cannot exceed 65,535 bytes. Its default is
        126 bytes. It may optionally be followed by
        a slash (/), which delimits lrecl from bsize.

NB   specifies that the file type to be allocated
        is nonbuffered indexed.

ITAM   specifies that the device to be allocated is
        a data communications device.

Functional Details:

If the file to be deleted by the XALLOCATE command does not exist, no error will occur. A file can be deleted and reallocated only if it is not currently assigned to a task, and its write and read protection keys are 0 (X'0000').

To assign an index file, sufficient room must exist in system space for three buffers (two data buffers and one index buffer), each of the stated size. Therefore, if bsize is very large, the file may not be assignable in some memory-bound situations. At sysgen, a maximum block size parameter is established in the system, and bsize cannot exceed this constant.

To assign an EC or NB file, sufficient room must exist in system space to contain only a single index block of the stated size. The data blocks for EC and NB files are not buffered in system space and, thus, are not constrained to the sysgened block size.

Examples:

XAL PROG1.TSK,IN,80   allocates on the system volume an indexed volume named PROG1.TSK with a logical record length of 80 bytes. Because no other parameters are entered, the file will have a physical block size with a default value of one sector and an index block size also with a default value of one sector. The write and read protection keys for the file will default to zero.

XAL SYS:XFILE.DTA,EC   allocates on volume SYS an extendable contiguous file named XFILE.DTA with a default data block size of 64 sectors and a default index block size of 3 sectors. The file initially contains no records and has a record length of one sector (same as a contiguous file). If the file already exists, has keys of 0000, and is not currently assigned to any task, it is deleted before being allocated. If any of these conditions are not met, an error occurs.

```
XAL YFILE.DAT,NB,240/250/5    reallocates on the system  volume  |
                              a  nonbuffered indexed file named  |
                              YFILE.DAT  with  logical   record  |
                              length  of  240 bytes, data block  |
                              size of 250  sectors,  and  index  |
                              block  size  of  5  sectors.  The  |
                              file   initially   contains    no  |
                              records.                           |
```

Messages:

ALLO-ERR

> indicates that the allocation failed for reasons denoted by
> the TYPE field. See Appendix B for possible entries in the |
> TYPE field. |

DELE-ERR

> indicates an error occurred while trying to delete a file.
> See Appendix B for possible entries in the TYPE field. |

FD-ERR

> indicates that an invalid fd is specified.

NODA-ERR

> indicates that direct access support was not included in the
> system at sysgen.

NOPR-ERR

> indicates that a required operand is missing.

PARM-ERR

> indicates that an operand syntax error occurred.

```
-------------
| XDELETE   |
-------------
```

3.61 XDELETE COMMAND

The XDELETE command is used to delete one or more files. If the
file does not exist, no error is generated.

Format:

    XDELETE fd$_1$ [,fd$_2$,...,fd$_n$]

Parameter:

    fd   is the file descriptor to be deleted.

Functional Details:

A file can be deleted only if it is not currently assigned to a
task and its write and read protection keys are 0 (X'0000').

Example:

    XDEL FIXD:OS323240.817,RADPROC.FTN

Messages:

DELE-ERR

    indicates that delete failed for the reason noted in TYPE
    field.   See Appendix B for possible entries in the TYPE
    field.

FD-ERR

indicates an invalid fd.

NODA-ERR

indicates that direct access support was not included in the operating system.

NOPR-ERR

indicates that no operand is specified.

CHAPTER 4
SYSTEM ERROR HANDLING

4.1 ERROR TYPES

There are three kinds of system level error conditions that can occur during operation of OS/32:

● Recoverable errors

● Abnormal task termination

● System failures

A recoverable error occurs when the system detects a condition that requires operator intervention in order to continue. In this case, the system prints a message describing the condition (e.g., PARM-ERR). The operator can then issue a command to correct the error. See Chapter 3 for the messages output by OS/32 while processing operator commands.

An abnormal task termination occurs when a malfunction is detected during execution of a user task (u-task), and the system cannot continue executing the task without destroying the system or user information. At this point, the system prints a message that describes the error detected (e.g., ILLEGAL INSTRUCTION, ILLEGAL SVC), and the location of the instruction causing the error. The u-task is then paused. The operator can correct the error condition and use the CONTINUE command to proceed, or the task can be aborted with a CANCEL command.

A system failure occurs when a hardware or software malfunction is detected during execution of system code, and the system cannot proceed without running the risk of destroying information, either on some peripheral device or in memory. At this point, the system displays a crash code on the display panel and the system console. It also displays on the system console a program status word (PSW) status and location of the crash condition; it then stops. System crash codes are summarized in Appendix D.

4.2 SYSTEM FAILURE RECOVERY

The system failure handler is designed to minimize destruction of user data when a failure occurs and to facilitate system debugging. The following steps must be accomplished to retrieve the information needed to determine the cause of the failure. Steps 3 and 4 must be accomplished whenever any failure occurs.

1. Record the system failure code and the sequence of the operator commands leading to the failure. This information can be obtained as follows:

   - Failure code - This code is displayed on the console display panel and is also stored in location SPT.CRSH in the system pointer table (SPT). System crash codes are summarized in Appendix D.

   - Operator command - If the console is a video display unit (VDU) device, note the information remaining on the screen. If the console is a hard copy device or LOG was set to a hard copy device, all information logged since the last operating system initialization should be preserved.

2. When possible, obtain a panic dump. Refer to Section 4.6.

3. Reload the operating system. Do NOT restart at location X'60'.

4. If any direct access devices were marked on at the time of the failure, run the FASTCHEK utility. See the OS/32 FASTCHEK Reference Manual.

5. To interpret the information produced by step 2, run the OS/32 dump print utility. See the OS/32 System Support Utilities Reference Manual.

4.2.1 Submitting Problems for Investigation

When submitting this information to Perkin-Elmer, please include the following information:

- PSW and location counter (LOC) at time of failure

- Panic dump tape

- Configuration statements and link edit map of system

- Console log prior to failure

- Time of failure

- Description of the failure including any system messages

- Conditions surrounding the failure

4.3 POWER FAIL/RESTORE

When a power fail is detected by the processor, an interrupt occurs and OS/32 saves the registers and prepares itself for another interrupt when power is restored. The following message appears on the console when power is restored:

POWER RESTORE - RESET PERIPHERALS AND ENTER GO

After the message is printed, operator intervention may be required to reset any devices that might be in an offline or write-protected state as a result of the power failure. The operator intervention requirement is a source system generation (sysgen) option. If operator intervention is required, the operator can continue the power restoration process by entering GO followed by a carriage return on the system console.

Upon completion of the power restoration process, the following message is output to the system console:

POWER FAILURE AT mm:dd:yy,hh:mm:ss

The date and time are displayed in the format specified at sysgen time. All nondirect access input/output (I/O) operations in progress at the time the power failed are aborted, and an error status is returned to the task. Direct access I/O operations are retried when power is restored.

The power restoration module schedules power restoration traps for those tasks that require such knowledge and have the power restoration trap enable bit set in the current task status word (TSW). When there is a power outage, a TSW swap occurs when power is restored, and all tasks with the power restoration trap bit enabled enter the user-defined power restoration trap routine. All other tasks are paused with the message:

taskid:TASK PAUSED

Refer to the OS/32 System Generation/32 (SYSGEN/32) Reference
Manual for information on eliminating operator intervention
requirements. Refer to the OS/32 Application Level Programmer
Reference Manual for further information on power
fail/restoration traps.

4.3.1 Shared Memory Power Failure

When a power failure occurs in a shared memory cabinet, the
following message is displayed on the system console of each
system utilizing shared memory:

SHARED MEMORY POWER FAILURE AT hh:mm:ss,mm:dd:yy

The time and date are displayed in the format selected at sysgen
time.

Tasks that attempt to read an area of shared memory will receive
the following error message and will be paused:

NONCONFIGURED MEMORY

4.4 DISK INPUT/OUTPUT (I/O) ERRORS

Certain disk related I/O errors and bit map errors indicate that
the integrity of the disk volume is in doubt. When possible,
OS/32 warns the operator of disk related errors by issuing the
following messages:

BIT MAP ERROR ON voln: MARK OFF AND CHECK

I/O ERROR ON voln: MARK OFF AND CHECK

In the event of these errors, the operator should mark off the
disk volume as soon as possible and execute the FASTCHEK utility.

4.4.1 Secondary Directory Overflow Error

When the secondary directory becomes full, the operator is
informed by the following message:

CDIR FULL - devn

The device mnemonic of the disk is devn. Refer to the MARK
command description in Chapter 3 for further information and
action necessary in this situation.

4.5 SYSTEM SHUTDOWN AND RESTART

In disk oriented OS/32 systems, it is necessary that the system be shut down or restarted in an orderly fashion to ensure the integrity of the disk volumes in use. In particular, before shutting down or restarting OS/32, the operator should:

● cancel and delete (make nonresident) all tasks, and

● mark all disk devices offline. ·

If the system is reloaded as a result of a system failure, the FASTCHEK utility should be used to restore the integrity of the data on all disk volumes online at the time of the failure.

NOTE

If a system fails, it must not be restarted at X'60'. It must be reloaded to ensure system integrity.

4.6 OS/32 PANIC DUMP PROCEDURE

The panic dump procedure allows the user to copy the entire contents of memory to a user-specified magnetic tape or disk device. Panic dumps should be performed immediately following a system failure due to a malfunction in:

● hardware,

● system software,

● a user-written driver, or

● areas in the operating system modified by the user.

After a panic dump has been performed, the contents of the dump should be listed to a print device using the dump print utility. Refer to OS/32 Systems Support Utilities Reference Manual for instructions regarding the use of the dump print utility.

4.6.1 Electing to Perform a Panic Dump

To perform a panic dump of a system that has not crashed, start
executing at location X'70'. It is recommended that the PSW be
cleared before execution is started.

The user has the option to perform a panic dump whenever a system
failure occurs. The following message is displayed immediately
below the system crash code and status information display:

DO YOU WISH A SYSTEM DUMP? (Y/N)

If a panic dump is not desired, enter N (no) which causes the
system to halt. Manual recovery must then be performed using the
procedure outlined in Section 4.2. If a panic dump is desired,
enter Y (yes) which causes the following program id message to be
displayed on the system console:

OS/32 SYSTEM DUMP Rnn-uu

Where:

nn   is a decimal number indicating the revision
                        level, and

uu   is a decimal number indicating the update
                        level

Immediately following the program id, a menu containing the names
of the magnetic tape and disk devices with standard
configurations supported by the panic dump program is displayed
as follows:

DEVS
MG85
MGC5
MG62
DS67
C13R
OTHR

The default configurations for each device in the menu are
presented in Table 4-1.

TABLE 4-1   PANIC DUMP PROGRAM SUPPORTED DEVICES WITH
STANDARD CONFIGURATIONS

DEVICE NAME	DEVICE DESCRIPTION	DEVICE CODE	DEVICE ADDRESS	CONTROLLER ADDRESS	SELCH ADDRESS
MG85	800 bpi mag tape	40	85	N/A	F0
MGC5	1600 bpi mag tape	41	C5	N/A	F0
MG62	6250 bpi mag tape	44	85	N/A	F0
DS67	67Mb disk	35	FC	FB	F0
C13R	13.5Mb disk - CMD removable	3B	FC	FB	F0

The following prompt is displayed immediately following the
device menu:

DEVICE=dddd

Where:

dddd   is a 4-character string indicating one of the
                        supported devices listed in the device menu.
                        Standard configurations for these supported
                        devices are automatically represented in the
                        system as listed in Table 4-1. If a unique
                        configuration is desired for a device listed
                        in the menu, enter OTHR. The type of device
                        entered in response to the DEVICE= prompt
                        determines which of the following three prompt
                        sequences is issued.

                        When a device is specified, the panic dump
                        program compares the amount of free space
                        available on the device to the size of memory
                        to be dumped. If there is insufficient free
                        space on the device to hold the entire memory
                        dump, the following message is displayed:

                        DUMP WILL NOT FIT

                        The panic dump program then returns to the
                        DEVICE= prompt to allow the user to select a
                        different device.

● Mag Tape Device

   If a magnetic tape device with a standard configuration is specified in response to the DEVICE= prompt, the panic dump program begins execution. Memory contents are written to tape in 2kb records. When the dump is successfully completed, this message is displayed on the system console:

   SYSTEM DUMP COMPLETE

   The tape is immediately rewound, registers are restored, and the machine is halted. At this point, a manual recovery from the system failure should be performed. See Section 4.2. If the tape does not rewind, the dump was not successful. Ensure that the tape drive is functional, that the write ring is inserted, and that the heads are clean, and then restart the procedure.

● Disk Device

   If a disk device with a standard configuration is specified in response to the DEVICE= prompt, the following message is displayed:

   VOLUME=DUMP,FILE=filename.ext

Where:

   DUMP   is the name of the volume to which memory contents are to be dumped. If the requested disk device has a valid volume mounted with a name other than DUMP or whose directory structure is not consistent with the panic dump format, the following message is displayed:

                              VOL xxxx IS TO BE OVERWRITTEN (Y/N)

                              If Y (yes) is entered, the panic dump program generates a new volume descriptor and directory. Ensure that there are no files that are required for future use on the volume to be overwritten. Panic dump could overwrite these files and the data will not be accessible in any case.

If N (no) is entered, the panic dump program will return to the DEVICE= prompt. This allows the user to specify a different device or change the disk pack.

filename.ext   is a 1- to 8-character filename, followed by a 1- to 3-character extension for the file that will hold the memory dump contents. The panic dump program automatically assigns the specified file to account 0. After the file descriptor (fd) is entered, the panic dump program begins execution. If the file already exists, the following message is displayed:

ERROR...FILENAME EXISTS ON DUMP DISK

If an invalid character is entered as part of the filename, the following message is displayed:

ERROR...FILENAME ENTRY

● Other

A response of OTHR to the DEVICE= prompt allows the user to specify a nonstandard device configuration for a supported device. The following prompt sequence is displayed on the system console:

DEV #=aaa

Where:

aaa   is a 1- to 3-digit hexadecimal number indicating the I/O bus address of the device to which the dump contents are to be sent.

DEV CODE=cc

Where:

cc   is a 2-digit hexadecimal number indicating the code of the device to which the dump contents are to be sent. The device codes listed in Table 4-1 are the only valid device codes. The device code entered here identifies the type of device to which the dump contents are being written. The next prompt in the sequence is dependent upon the type of device indicated by the device code entry.

CTLR ADDR=nnn

Where:

nnn   is a 1- to 3-digit hexadecimal number indicating the I/O bus address of the controller address for the device indicated by the device code. This prompt is not displayed if the device is a magnetic tape device.

SLCH ADDR=sss

Where:

sss   is a 1- to 3-digit hexadecimal number indicating the I/O bus address of the selector channel (SELCH).

4.6.2 Disk Utilization

A single disk can hold no more than five panic dumps, depending upon the size of the disk and the size of memory to be dumped. If a disk directory already contains five dump entries and a sixth is attempted, the following message is displayed:

VOL xxxx IS TO BE OVERWRITTEN (Y/N)?

If the panic dump program determines that there is not enough disk space to hold an entire memory dump, the following message is displayed on the system console before the dump begins:

DUMP WILL NOT FIT
VOL xxxx IS TO BE OVERWRITTEN (Y/N)?

If Y is entered, the panic dump program generates a new volume descriptor and directory. If N is entered, the program returns to the DEVICE= prompt to allow the user to specify a different device or change the disk pack.

If Y is entered, the panic dump program reinitializes the disk specified as the storage disk and executes. Data on the disk will be overwritten. Therefore, before entering Y as a response to this prompt, ensure that there is no data to be saved for future use on the disk. Once the panic dump has been completed, the user can access the dump file through the operating system by marking the disk on, via the MARK ON PROTECT command, and then assigning the dump file, via the ASSIGN command. A panic dump cannot be accessed unless the disk has been run through FASTCHEK |
or is marked on, protected. In order to delete a dump file, the |
user must first run the FASTCHEK utility on the disk to rebuild |
the bit map. The panic dump routine can be restarted at X'70'
following an error condition.

4.6.3 Examples of Panic Dump Device Specifications

The following are examples of the panic dump prompt sequence for various devices. Underlining indicates user-supplied information.

- Writing the contents of a panic dump to an 800 bpi magnetic tape with a standard configuration:

```
        DO YOU WISH A SYSTEM DUMP (Y/N)? Y
        OS/32 SYSTEM DUMP R06-02                              |
        DEVS
        MG85
        MGC5
        MG62
        DS67
        C13R
        OTHR
        DEVICE = MG85
              .
              .
              .
        SYSTEM DUMP COMPLETE
```

- Writing the contents of a panic dump to a CDD   removable   disk
  with a standard configuration:

```
         DO YOU WISH A SYSTEM DUMP (Y/N)? Y
         OS/32 SYSTEM DUMP R06-02
         DEVS
         MG85
         MGC5
         MG62
         DS67
         C13R
         OTHR
         DEVICE = C13R

         VOLUME = DUMP,FILE=DUMP0928.001
                     .
                     .
                     .
         SYSTEM DUMP COMPLETE
```

- Writing the contents of a panic dump to an   800   bpi   magnetic
  tape device with a nonstandard configuration:

```
         DO YOU WISH A SYSTEM DUMP (Y/N)? Y
         OS/32 SYSTEM DUMP R06-02
         DEVS
         MG85
         MGC5
         MG62
         DS67
         C13R
         OTHR
         DEVICE = OTHR
         DEV # = 85
         DEV CODE = 40
         SLCH ADDR = F4
                     .
                     .
                     .
         SYSTEM DUMP COMPLETE
```

● Writing the contents of a panic dump to a 67Mb fixed disk with a nonstandard configuration:


```
        DO YOU WISH A SYSTEM DUMP (Y/N)? Y
        OS/32 SYSTEM DUMP R06-02
        DEVS
        MG85
        MGC5
        MG62
        DS67
        C13R
        OTHR
        DEVICE = OTHR
        DEV # = EC
        DEV CODE = 35
        CTRL ADDR = EB
        SLCH ADDR = FA
        VOLUME=DUMP,FILE=DUMP0928.001
                    .
                    .
                    .

        SYSTEM DUMP COMPLETE
```


4.6.4 Panic Dump Messages

ERROR...FILENAME EXISTS ON DUMP DISC

   The filename entered by the user for the dump file already exists on the specified disk.

IOERROR DU STATUS=XX

   The user-specified device is unavailable. XX is the hexadecimal hardware status sent from the device. Refer to the appropriate device hardware manual for an explanation of the status setting.

IOERROR CNFG STATUS=XX

   An I/O error occurred because the hardware configuration does not correspond with that specified by the user. XX is the hexadecimal hardware status sent from the device. Refer to the appropriate device hardware manual for an explanation of the status setting.

IOERROR UNRE STATUS=XX

   An unrecoverable I/O error has occurred. XX is the hexadecimal hardware status sent from the device. Refer to the appropriate device hardware manual for an explanation of the status setting.

CHAPTER 5
THE COMMAND SUBSTITUTION SYSTEM (CSS)

5.1 INTRODUCTION

The OS/32 CSS is a versatile command language that can be used to control anything from a single batch stream to complex macro commands. CSS has many of the characteristics of a programming language. Its structure, however, allows simple functions to be performed without complete familiarity with CSS. This chapter is intended as a guide for the user, from the beginning stages through the more advanced stages of CSS use. It is assumed that the user is familiar with the functions of the OS/32 operator commands, or at least with their most common forms, described in Chapter 3 of this manual. While CSS is active, the system console remains available for operator commands and task input/output (I/O).

5.2 COMMAND SUBSTITUTION SYSTEM (CSS) FILES

A CSS file is a series of OS/32 operator commands. Every operator command can be executed from a CSS file. In addition, a number of special CSS commands can be executed. The special CSS commands (except SET CODE) begin with a dollar sign, making them easy to identify. The series of operator commands contained in a CSS file exists on media such as cards, disk, or magnetic tape. When the CSS file is called, the commands contained within the file are executed in order. When the special CSS command $EXIT is executed, CSS execution is stopped and control is returned to the caller. CSS files can be called by the operator or other CSS files.

If the CSS procedure is to be contained in a disk file, the CSS filename specified must be different from all standard operator commands and mnemonics, including any abbreviations. Thus, these are valid CSS filenames:

    ABC.CSS

    XYZ.CSS

    AB.CSS

The following filenames would be valid only when accompanied by
the .CSS extension:

AL.CSS
RW.CSS
Pl.CSS

If the .CSS extensions are omitted from the above filenames, they
would become invalid for the following reasons:

● AL is the abbreviation for the ALLOCATE command.

● RW is an abbreviation for the REWIND command.

● P is the abbreviation for the PAUSE command. Mnemonic scan
  stops at the first nonalphabetic character.

These filenames can be made valid by prefixing the call to CSS
with a volume name. For example:

M300:AL
M301:RW
MTM:Pl

5.2.1 Calling Command Substitution System (CSS) Files

The CSS file is read from a source input device; e.g., from a
card reader, magnetic tape, etc. The device being used must be
online. CSS files may also exist as direct access files. To
call the CSS file, the operator enters the name of the device or
disk file in the same way as a command is entered. For example,
if the file is on cards, and the card reader is named CARD, the
operator enters the following and the card file is executed:

CARD:

5.2.2 One Command Substitution System (CSS) File Calling Another

To OS/32, a CSS file is the same as the system operator.
Therefore, one CSS file can call another. However, when too many
CSS files are active, the command processor buffers may fill and
a new CSS file cannot be called. The maximum nesting depth is
set at system generation (sysgen). If one CSS file tries to call
another, exceeding the designated depth, the system produces the
following error message and CSS processing ceases:

LVL-ERR

When one CSS file calls another, the first file is still active.
When the second file finishes processing, execution continues in
the first CSS file at the command following the call to the
second CSS file.

5.3 USING COMMAND SUBSTITUTION SYSTEM (CSS) FOR BATCH CONTROL

CSS can be used to control a batch job. This allows the user to
prepare a sequence of operator commands and the data for the job,
and then to enter the entire job stream to OS/32 through CSS.

5.3.1 Job Control Decks

The following job control deck illustrates the use of CSS to
control a batch job from the card reader (CR:):


```
    LOAD .BG,CAL32.TSK              Load     assembler  from     file
                                      CAL32.TSK
    TASK .BG
    ASSIGN 1,CR:                    Assign  source  input  to  this
                                      file
    ASSIGN 2,MAG1:
    ASSIGN 3,PR:
    TEMPFILE 4,IN,80
    START ,TARGT=32,SCRAT          Start assembler
        .
        . (source program)
        .
    REWIND MAG1:
    BUILD LNK.CMD
    MAP PR:,ADDR,ALPH,XREF
    INCL MAG1:
    BUILD PROG.TSK                 Load module is now built
    END
    ENDB
    LOAD .BG,LINK                  Load Link
    TASK .BG
    START ,COM=LNK.CMD,L=CON:      Start Link
    LOAD .BG,PROG.TSK              Load new module
    TASK .BG
    START                         Run program
    $EXIT                         End job
```


The first statements load the CAL32 assembler, assign logical
units to devices and files, and start the assembly process.
Following these statements are the source statements of the
program to be assembled.   Next, are the command statements to
load Link.   The START command begins execution of Link.   It is
followed by a series of Link commands, terminated by the Link END
command.   The Link commands are followed by a LOAD command that
loads the module Link produced and runs the program.   Finally,
the $EXIT command returns control to the operator.

5.3.2 Separation of Jobs

When any error occurs in processing a CSS file, CSS processing stops, and control is returned to the system console. This may be desirable in some cases, but not when batch jobs are being run. If there are several sets of job control statements in the CSS file (for example, in the card reader), the faulty termination of one job must not cause all jobs to be aborted. The CSS commands $JOB and $TERMJOB are used to confine the effects of errors to a single job. Each job control deck may start with a $JOB statement and end with a $TERMJOB statement. Then, if an error in a given job is detected, the CSS processor skips all commands until a $TERMJOB is found, and resumes normal CSS processing.

A typical batch deck, consisting of several jobs, might look like this:

```
$JOB                          Start of first job
    .
    .  (first job control deck)
    .
$TERMJOB                      End of first job

$JOB
    .
    .  (second job control deck)
    .
$TERMJOB                      End of second job

$JOB
    .
    .  (further job control decks)
    .
$TERMJOB
$EXIT                         End of batch stream
```

NOTE

It is not permissible to nest jobs.

5.3.3 Program Pauses and Other Interactions

It is not good practice to let CSS processing continue while a program called by CSS is running. In any job control deck, the job steps are sequential. When a program is started, it should run to completion before more CSS statements are executed. Otherwise, the CSS processor might try to perform device assignments for the second job step before the first one is finished.

When a background program pauses, CSS processing does not resume. Instead, a pause message is routed to the system console. The operator is responsible for correcting the error and issuing a CONTINUE command. As far as CSS is concerned, the program is still active, whether or not it has paused. If the program goes to end of task or if it is cancelled by the operator, CSS processing resumes. This procedure assumes that programs pause only due to an error or due to the occurrence of some abnormal condition for which the program is unable to take corrective action by itself. Thus, operator intervention is required. Programs that pause arbitrarily or common assembly language (CAL) assemblies with PAUSE or PPAUS statements in their source decks cannot be properly handled under CSS batch control.

5.3.4 $PAUSE, $CONTINUE, and $WAIT Commands

The $PAUSE command suspends CSS, allowing time for the operator to respond to an external event; e.g., mount a tape. The $PAUSE command is valid only from CSS and cannot be entered from the system console. If entered, the following message is displayed:

PAUS-ERR

The $CONTINUE command continues a CSS suspended by a $PAUSE or $WAIT. The $WAIT command suspends a CSS for a specified period of time. It is valid only in CSS. If the user does not want to wait the specified period of time, the $CONTINUE command can be entered to continue the CSS.

The formats for these commands are:

$PAUSE

$CONTINUE

$$\$WAIT \left[\left\{ \begin{array}{c} n \\ 1 \end{array} \right\} \right]$$

The parameter n in the WAIT command is a decimal number from 1 to 900 indicating the number of seconds that CSS is suspended. If the parameter is omitted, the default is 1 second.

5.4 USING COMMAND SUBSTITUTION SYSTEM (CSS) TO AVOID REPETITIOUS ACTIONS

CSS is not used for batch control only. It also can be used to avoid lengthy, repetitious operator instructions. Assume that at a given installation, all CAL assemblies are done with a standard set of logical unit (lu) assignments. Command statements for these assignments are made into a CSS file, and that CSS file is put on disk and used before each CAL assembly. For example, assume a disk file named CALASIGN.CSS contains the following commands:

```
    CLOSE 1,2,3,4,5,6,7,8,9
    XALLOCATE CALSCRAT.TMP,IN,80
    XALLOCATE CROSSREF.TMP,IN,256
    XALLOCATE SYMDUMP.TMP,IN,256
    XALLOCATE SQUEEZ.TMP,IN,256
    XALLOCATE ERRLST.TMP,IN,80
    ASSIGN 1,CR:
    ASSIGN 2,MAG1:
    ASSIGN 3,PR:
    ASSIGN 4,CALSCRAT.TMP
    ASSIGN 5,CROSSREF.TMP
    ASSIGN 6,SYMDUMP.TMP
    ASSIGN 7,SORCELIB.CAL
    ASSIGN 8,SQUEEZE.TMP
    ASSIGN 9,ERRLST.TMP
    $EXIT
```

The operator places the source program deck in the card reader device (CR:), mounts an output tape on the magnetic tape device (MAG1:), and types:

```
    LOAD .BG,CAL32
    TA .BG
    CALASIGN
    START ,SCRAT,CROSS,ERLST
```

and the assembly proceeds. The LOAD and START commands also can be put on the CSS file. This procedure reduces operator intervention and the possibility of error.

CSS can be used to modify OS/32 with patch information after system startup. If an installation has a list of patches to be entered, the patches are prepared on a card deck, with verification information, as in the following example:

```
    BIAS 0
    MODIFY 1FE0,220,80A,4300,1ED4
    EXAMINE 1FE0,4
    $EXIT
```

Immediately upon system startup, this deck is put in the card
reader, and the command CR: is entered from the console. The
CSS file is then read, and the patches are made to the system
without the problem of typing errors. The EXAMINE command
displays the contents of the patch area following the
modification.

5.5   USING COMMAND SUBSTITUTION SYSTEM (CSS) TO BUILD COMPLEX
         COMMANDS

CSS also is used as a system macro command language to build
complex commands. The basic set of commands provided in the
command processor is sufficient to perform any function, but may
require operator interaction.

5.5.1   Passing Arguments to Command Substitution System
            (CSS) Files

The CAL assignment example given in Section 5.4 does not work if
some of the devices vary from assembly to assembly. When
executed, file CALASIGN always assigns lul to CR:, lu2 to MAG1:,
etc.

To make assignments more flexible, it is necessary to pass
arguments to a CSS file. This is accomplished by using the
character sequence @n. Whenever the sequence @n (where n is an
integer greater than or equal to 0) is detected in a CSS command,
the CSS processor uses the n argument in the CSS calling
statement. For example, assume that CSS file ASINE123.CSS
contains the following commands:

        ASSIGN 1,@1
        ASSIGN 2,@2
        ASSIGN 3,@3
        $EXIT

and the operator types:

        ASINE123 CR:,MAG1:,PR:

The CSS file is executed as though it contains:

        ASSIGN 1,CR:
        ASSIGN 2,MAG1:
        ASSIGN 3,PR:
        $EXIT

The first argument of the calling statement is CR:, and every occurrence of the sequence @1 in the CSS file is replaced with CR:. An argument is always terminated with a comma, semicolon, or carriage return. Therefore, the second argument is MAG1:, and the third argument is PR:. In the previous example, CALASIGN can be turned into a generalized ASSEMBLE CSS as follows:

```
LOAD .BG,CAL32
TASK .BG
CLOSE 1,2,3,4,5,6,7,8
XALLOCATE CALSCRAT.TMP,IN,80
XALLOCATE CROSSREF.TMP,IN,256
XALLOCATE SYMDUMP.TMP,IN,256
XALLOCATE SQUEEZE.TMP,IN,256
XALLOCATE ERRLST.TMP,IN,80
ASSIGN 1,@1                      ;*VARIABLE INPUT
ASSIGN 2,@2                      ;*VARIABLE OUTPUT
ASSIGN 3,@3                      ;*VARIABLE LISTING
ASSIGN 4,CALSCRAT.TMP
ASSIGN 5,CROSSREF.TMP
ASSIGN 6,SYMDUMP.TMP
ASSIGN 7,@4                      ;*VARIABLE SOURCE LIBRARY
ASSIGN 8,SQUEEZE.TMP
ASSIGN 9,ERRLST.TMP
START ,SCRAT,CROSS,ERLST
$EXIT
```

This file can be called as follows:

```
ASSEMBLE CR:,MAG1:,PR:,SORCELIB.CAL
```

The first argument is the input device; the second, the output device; the third, the listing device; and the fourth, the source library device. Note that a semicolon may be used to separate one CSS command statement from another; an asterisk in the first position of a CSS statement causes it to be treated as a comment statement.

5.5.2 Device Independent Job Control Decks

The example in Section 5.5.1 assumes that the CSS file is from a device name. It would not work properly if the device were changed to MAG2: since the CSS statements specifically assign lul of CAL32 to a device name CR:. To create a device independent job control deck, use the characters @0 whenever referring to the CSS file itself. The characters @0 mean "this device or file". The ASINE123 example must be changed as follows:

```
LOAD .BG,CAL32
TASK .BG
ASSIGN 1,@0
ASSIGN 2,MAG1:
ASSIGN 3,PR:
$EXIT
```

To start the CSS job stream, issue this from the system console:

```
CR:
```

lul is assigned to CR:.

5.5.3 Testing Arguments for Existence

Assume that the operator wants to assemble the previous program without using a source library device.

Example:
)

```
ASSEMBLE CR:,MAG4:,PR:
```

The fourth argument is missing. A missing argument is considered to be a null string, that is, a sequence of no characters. Therefore, the command ASSIGN 7,@4 is executed as:

```
ASSIGN 7,
```

This is an illegal command, and execution of the CSS file aborts.

It is possible to test whether or not an argument exists. This is done with one of the special CSS commands, $IFNULL or $IFNNULL. The $IFNULL command tests to see whether an argument does not exist. The $IFNNULL command tests to see whether an argument does exist.

If the tested condition is true, the CSS processor continues to execute statements until a $ELSE is found. At that time, the CSS processor skips until the corresponding special command $ENDC (end conditional) is found. If a $ELSE is not present before the corresponding $ENDC, the CSS processor continues to execute statements.

If the tested condition is not true, the CSS processor skips all statements until command $ELSE is found, at which time the CSS processor resumes executing statements. If a $ELSE is not present before the corresponding $ENDC, statement execution resumes after the $ENDC. Therefore, the command:

        ASSIGN 7,@4

can be replaced with the following:

        $IFNNULL @4   Is there a source library?
        ASSIGN 7,@4   If so, assign it to lu7
        $ELSE   If not,
        ASSIGN 7,NULL: assign lu7 to the null device
        $ENDC   End of conditional

If, when skipping, the CSS processor attempts to skip beyond a $TERMJOB or end of file (EOF), an error occurs. A combination of parameters can be tested simultaneously. For example:

        $IFNULL @1@2@3

| This performs similarly to the three statements:

| $IFNULL @1
| $IFNULL @2
| $IFNULL @3

| providing a logical AND of the truth value of the statements. If any of the three @n parameters are not null, the test results in a false condition.

CSS also can be used to set up default assignments. For example, assume the list device is normally the printer (PR:). The following sequence is put into the ASSEMBLE.CSS file:

```
$IFNULL @3            Is    there    a    list    device
                       specified?
ASSIGN 3,PR:          If not, use PR:
$ELSE                 If there is,
ASSIGN 3,@3            assign it to lu3
$ENDC
```

If the operator then enters:

```
ASSEMBLE MAG2:,MAG3:,,MAG4:
```

argument 3 is null; therefore, the listing is assigned to the printer (PR:).

5.5.4 Testing Files for Existence

It is often desirable to allocate files, using commands in a CSS file. Assume the ASSEMBLE.CSS file previously developed requires a number of scratch files. Then:

● if the scratch file already exists, it is an error to try to reallocate it, and the CSS file aborts; or

● if the file is not allocated and it does not exist, the ASSIGN command is in error, and the CSS file aborts.

To solve this problem, a facility is provided to allow the CSS file to test for the existence of certain files. This is done with the $IFX (if file exists) and $IFNX (if file does not exist) commands. These special commands work in the same way as the $IFNULL and $IFNNULL commands. That is, if the tested condition is true, CSS commands are executed up to the corresponding $ELSE or $ENDC. If the test proves false, subsequent commands are skipped up to the corresponding $ELSE or $ENDC. If the program skips beyond a $TERMJOB, an EOF error occurs. For example:

```
$IFNX CALSCRAT.TMP            Does CALSCRAT.TMP exist?
ALLOCATE CALSCRAT.TMP,IN,80   No, allocate it
$ENDC                         End of conditional
```

This sequence checks to see if the temporary file already exists and allocates it if it does not exist.

CSS can be used to test for the existence of a file passed to the CSS file as an argument. For example, consider this CSS file:

```
LOAD .BG,TASK1              Load a task named TASK1
TASK .BG
$IFNX @1                   If the file does not exist,
  ASSIGN 1,CON:             input source from CON:
$ELSE                      If the file does exist,
  ASSIGN 1,@1               use it for input
$ENDC
ALLOCATE T1.TMP,IN,80      Allocate an output file
ASSIGN 2,T1.TMP,ERW        Assign for exclusive read-write
ASSIGN 5,CON:              Assign command input to CON:
START                      Start the editor
$IFX @1                    If the CON:  was not used,
  RENAME @1,T1.OLD           rename the input file
RENAME T1.TMP,@1           Rename the output file
$ENDC
$EXIT
```

If argument1 is the name of an existing file, that file is processed. The output file is named T1.TMP, but after execution is completed, the output file is given the name of the input file, and the input file is renamed T1.OLD. If argument1 specifies a file that does not exist yet, the input device is assumed to be CON:. The RENAME command is not applicable to nondirect access devices.

5.5.5 Multi-Level Parameter Passing

Use of the multiple @ notation is permitted to reference higher level parameters; i.e., in a CSS file that is called by other CSS files, but not by the system operator, it is possible to refer to parameters passed in both the current CSS file and its caller. This is done by using more than one @ sign to indicate the parameter number. For example:

@1   is the first parameter passed to the current file.

@@4   is the fourth parameter passed to the caller of the current file.

@@@7   is the seventh parameter passed to the caller's caller.

If the number of @ signs exceeds the current nesting level, the null string is returned. This can be tested for by using the $IFNULL command. The multiple @ sign convention can also be used with the special code @0. The @0 sign refers to the identifier of the current file, @@0 refers to the identifier of the caller of the current file, and @@@0 refers to the caller's caller, etc.

5.5.6 Making Files Caller-Independent

The previous example works properly only if the console device is called CON: and if the CSS procedure is called from the console | device. It can be modified to work with any caller. It is | desirable to be able to call a CSS procedure from either a job control deck or from a console device.

A file can be made caller-independent by replacing references to the caller with the character sequence @@0, referring to the filename of the caller. Therefore, the changes to the above procedure are:

.
.
.

```
$IFNX @1                          If the file does not exist,
ASSIGN 1,@@0                        input from the caller
     .
     .
     .
ASSIGN 5,@@0                      Assign command input to caller
```

5.5.7 End of Task Codes and Error Handling

When a program called by CSS detects an error and abnormally terminates, CSS should take special action. In a job control deck that performs a compilation, assembly, creation of a load module, loading, and execution of a program, it is useless to continue the process if the first step (the compilation) terminates in error. The CSS must be informed of the erroneous end of task. This is done through the end of task code. Each program, on terminating, returns an end of task code that can be used to show why the program terminated. This end of task code is a number defined by the programmer in the supervisor call (SVC) 3 call that terminates the program. For details on SVC 3, see the OS/32 Supervisor Call (SVC) Reference Manual. An end of task code of 0 means the program terminated properly with no errors. A nonzero end of task code means the program terminated abnormally. An end of task code of 255 means the task was cancelled by the system operator.

CSS files can test the end of task code with a set of special CSS commands. These are:

```
$IFE   n                         Equal
$IFG   n                         Greater
$IFL   n                         Less
$IFNE  n                         Not equal
$IFNG  n                         Not greater
$IFNL  n                         Not less
```

Examples of these commands are as follows:

```
$IFE 14                          If end of task code = 14
$IFG 0                           If end of task code > 0
$IFL 255                         If end of task code < 255
$IFNE 1                          If end of task code ≠ 1
$IFNG @3                         If end of task code < the third
                                   argument
$IFNL 12840                      If end of task code > 12840
```

If the tested condition is true, CSS continues to execute commands until a $ELSE or $ENDC is found. If the condition is false, CSS skips all statements until a $ELSE or $ENDC is found. In such skipping, attempts to skip beyond a $TERMJOB or to EOF cause an error.

The special command $SKIP causes CSS to skip to the next $TERMJOB command unconditionally, if it is inside a job control deck. This has the same effect as any error detected in a CSS statement. For example:

```
LOAD .BG,CAL32                   Load the assembler
TASK .BG
  .
  .
  .
START                            Start it
$IFNE 0                          Any errors?
$SKIP                            If yes, skip further processing
$ENDC                            Otherwise, continue
```

This is the simplest way to handle errors. If the program has multiple error end of task codes, each with a distinct meaning, the CSS file may inform the operator of the nature of the problem, as follows:

```
LOAD .BG,PROG              Load the program
TASK .BG
START                      Start it
$IFNE 0                    If any errors
$IFNE 1                    RC=2 means fatal error
$SKIP                      So skip
$ENDC                      But RC=1 means nonfatal error
$COPY
*NONFATAL ERROR DETECTED
$NOCOPY
$ENDC                      Continue after logging message
```

Notice the nesting of the $IFNE...$ENDC sequences. Nesting of any $IF tests is permitted, with no restrictions on nesting depth.

A CSS file may test or set the end of task code. This is possible as CSS is active when the controlled program is inactive, and therefore, the end of task code is not used asynchronously. The end of task code is set using the command:

```
SET CODE n
```

which sets the end of task code to n, which is any number from 0 through 255.

The ability of CSS to set the end of task code may be used by one part of a CSS file to signal another, provided there are no programs started between the setting and testing of the end of task code.

5.5.8 Logging Messages to the Console

The special CSS commands $COPY and $NOCOPY or $WRITE allow the CSS file to log messages to the system console and the system log file, if present. Normally, CSS should be in $NOCOPY mode. Under these conditions, CSS commands are executed but are not printed on the console or on the log file. If the CSS command $COPY is executed, a copy of all CSS statements executed up to and including the next $NOCOPY is printed at the console, on the log file, or both. The choice of routing is set up by the system console operator with the SET LOG command, and CSS must not attempt to override this choice.

Since comments are printed, as well as commands, an easy way to log a message is:

```
$COPY
*THIS IS A MESSAGE FOR THE OPERATOR
$NOCOPY
```

The easiest way to log a message is:

```
$WRITE THIS IS A MESSAGE FOR THE OPERATOR
```

5.6 CREATING COMMAND SUBSTITUTION SYSTEM (CSS) FILES ON DISK

CSS files can be created on disk in one of two ways:

- OS/32 Edit or OS/32 Source Updater Utility programs can be used.

- BUILD and ENDB commands can be used to create a CSS file directly from the system console.

The BUILD command is used as follows:

```
BUILD filename.ext
```

This command causes an indexed file named filename.ext to be allocated, with a logical record length equal to the buffer length determined at sysgen, and with protection keys of zero. If the fd already exists, it is deleted and reallocated unless APPEND is specified. In this case, the new statements are appended to the existing file. The console enters into data entry mode so that data can be written to the allocated file. Every line typed at the console, from that time until an ENDB command is found, is written to the file being built. No parameter substitution is possible with the BUILD...ENDB sequence. This data may include any text, including commands, @n sequences, etc. The command processor does not attempt to execute any commands contained in the text, nor does it attempt to expand @n sequences. The only item that cannot be written to a file being built is a line with the characters ENDB as the first command on the input line. When an ENDB command is read, the console reverts to its normal command mode. When the console is in the build data entry mode, the prompt issued at the console is:

```
.CMDP>
```

To build a CSS file named EDIT.CSS, this sequence is entered:

```
BUILD EDIT
.CMDP>*THIS IS A CSS FILE NAMED EDIT.CSS
.CMDP>LO .BG,EDIT32;TA.BG
.CMDP>ST
.CMDP>EXIT
.CMDP>ENDB
```

This CSS file is now ready for execution. It can be called as follows:

```
FRED.CSS
```

or

```
FRED
```

since the CSS processor uses the default extension .CSS if the system operator does not enter an extension. The BUILD command also assumes the extension .CSS if the specified name does not include an extension.

A $BUILD...$ENDB sequence can be nested inside a BUILD...ENDB pair.

5.7 BUILDING TASK CONTROL FILES

Although it is possible to embed task control commands within a CSS file, certain problems can occur when using this technique. In this example the statements between START and $EXIT are not CSS commands but are commands to Link.

```
LOAD .BG,LINK              Load Link
TASK .BG
START ,COMM=@0,L=CON:
INCL PROG.OBJ
MAP PR:,XREF
BU PROG.TSK
END
$EXIT
```

The START command activates Link which then reads from the beginning of the CSS file. An error occurs as Link tries to process the LOAD command. Link is ready to execute another Link command and does not recognize LOAD.

To prevent these problems, the BUILD and ENDB commands can be used. For example:

```
      $BUILD COMMAND.TMP                 Build command file
      INCL PROG.OBJ
      MAP PR:,XREF
      BUILD PROG.TSK                     Link's BUILD command
      END
      $ENDB
      LOAD .BG,LINK                      Load Link
      TASK .BG
      START ,COMM=COMMAND.TMP,L=CON:     Start Link
      DELETE COMMAND.TMP                 Delete command file
      $EXIT
```

This sequence builds a separate task control file called COMMAND.TMP and assigns it to the Link command input lu. The control file is then deleted from the disk before the CSS file terminates.

The $BUILD and $ENDB commands allow parameter substitution within the file being built. If the specified fd does not already exist, it is created as it is with the BUILD command. The account number must be zero if specified. If the fd already exists, it is deleted and reallocated.

As with BUILD, no nesting of $BUILD is possible. Nesting will result in an invalid CSS file. $BUILD and BUILD must appear last on the input line. If any additional commands are entered on the line, they are ignored. $ENDB and ENDB must be the first commands on the input line. Commands following $ENDB and ENDB are executed. A BUILD...ENDB sequence can be nested within a $BUILD...$ENDB pair.

Using the feature of parameter substitution, the following example shows how the fd PROG.OBJ can be passed as a parameter from the caller.

```
      $BUILD COMMAND.TMP                 Build command file
      INCL @1
      MAP PR:,XREF
      BUILD PROG.TSK                     Link's BUILD command
      END
      $ENDB
      LOAD .BG,LINK                      Load Link
      TASK .BG
      START ,COMM=COMMAND.TMP,L=CON:     Start Link
      DELETE COMMAND.TMP                 Delete command file
      $EXIT
```

Assuming the CSS file is named LNKBUILD.CSS, it could be invoked
as follows:

LNKBUILD PROG.OBJ |

5.8   EXITING FROM COMMAND SUBSTITUTION SYSTEM (CSS) FILES

Three commands are provided for exiting from CSS files: $EXIT,
$TRANSFER, and $CLEAR. $EXIT causes control to return to the
place from which the CSS file was called.   Control returns either
to the console or to a higher level CSS file. $EXIT must
terminate each CSS file called. $CLEAR causes control to return
to the console, immediately terminating all CSS activity.   The
$TRANSFER command causes the program to exit from the current CSS
with all other levels closed and to enter into a new CSS.   The
formats of the commands are:

    $EXIT

    $CLEAR

    $TRANSFER NEWCSS paraml, param2,...

5.9   USING STANDARD FILE EXTENSIONS

In a disk-based system, standard file extensions can be used to
save the system operator work and to allow CSS files to perform
sophisticated functions.   The concatenation facility of CSS
allows the use of standard extensions.   In an example of FORTRAN
compilation, useful standard file extensions are:

    .FTN   FORTRAN source
    .CAL   CAL source
    .LST   List file
    .ERR   Error listing file

A file COMPILE.CSS can be created to control   compilation.   For
example:


```
*COMPILE CSS FILE
LOAD .BG,FORTRAN             Load the compiler
ASSIGN 1,@1.FTN             Assign FORTRAN input
ASSIGN 2,@1.CAL,ERW          Assign CAL output
ASSIGN 3,@1.LST,EWO          Assign listing (append)
ASSIGN 7,@1.ERR             Assign error message file
START                        Start compiler
$EXIT                        And exit
```

This CSS procedure is called as follows:

    COMPILE

The COMPILE CSS procedure assumes the FORTRAN source is on a file
called progname.FTN and puts the output, listing, and error
messages onto files named progname.CAL, progname.LST, and
progname.ERR, respectively. This procedure assumes that all the
named files already exist.

5.10 INTERACTION OF COMMAND SUBSTITUTION SYSTEM (CSS) WITH FOREGROUND/BACKGROUND SYSTEMS

CSS is essentially a single-stream processor. Usually it is not
possible to write a CSS file that can fully control a complex
foreground/background system. Manual intervention by the system
operator is often required to control such a system. Foreground
systems are controlled, under normal circumstances, by intertask
communication (SVC 6) calls between foreground tasks. The system
operator is required to intervene only in abnormal cases. The
background system, however, must be controlled fairly often by
the operator, or by CSS files, if the background is being run in
a batch-like mode.

The background taskid (.BG) is used to control the coordination
between task execution and CSS execution. CSS is active only if
a CSS file is invoked when the background task is dormant. While
the background task is in any state other than dormant, CSS is
inactive. The state of foreground tasks has no effect on CSS
activity.

While CSS is active, the operator still can enter commands from
the system console. The execution of these commands may be
delayed while a CSS command is being executed; however, this
delay should not be excessive under normal circumstances. If a
CSS file is active, any attempt to call another CSS file from the
system console is rejected with a SEQuence-ERRor.

CSS files may affect foreground tasks, but extreme care should be
used, since a task in the foreground, once initiated, is no
longer under CSS control. A TASK command read from a CSS file
establishes the currently selected CSS task. Commands from the
console are not affected by TASK commands read from CSS, and CSS
is not affected by TASK commands input via the console. All task
related commands (Chapter 3) and CSS end of task code testing
encountered in a CSS file affect the currently selected CSS task.

When a CSS file is activated from the console, the currently selected CSS task is set equal to the currently selected task. If the currently selected CSS task is deleted from the system, any subsequent task-related or CSS end of task code testing commands are rejected with a task error (TASK-ERR).

5.11 COMMAND SUBSTITUTION SYSTEM (CSS) COMMAND SUMMARY

The following is a summary of CSS commands and their meanings:

BUILD fd [,APPEND]

> constructs a CSS file without parameter substitution.

ENDB   specifies end of BUILD.

SET CODE n   sets end of task code to n.

$BUILD fd   constructs a CSS file with parameter
[,APPEND]   substitution.

$CLEAR   returns control to console.

$CONTINUE   continues a CSS file that was suspended by a $PAUSE or $WAIT.

$COPY   produces a listing.

$ELSE   reverses the effect of $IF until a $ENDC corresponding to $IF is encountered.

$ENDB   specifies end of $BUILD.

$ENDC   delimits conditionals described above.

$EXIT   specifies exit from CSS file.

$IFE n   specifies that if end of task code equals n, continue executing commands; otherwise, skip to the corresponding $ELSE or $ENDC.

$IFG n   specifies that if end of task code is greater than n, continue executing commands; otherwise, skip to corresponding $ELSE or $ENDC.

$IFL n   specifies that if end of task code is less than n, continue executing commands; otherwise, skip to the corresponding $ELSE or $ENDC.

$IFNE n	specifies that if end of task code is not equal to n, continue executing commands; otherwise, skip to the corresponding $ELSE or $ENDC.
$IFNG n	specifies that if end of task code is not greater than n, continue executing commands; otherwise, skip to the corresponding $ELSE or $ENDC.
$IFNL n	specifies that if end of task code is not less than n, continue executing commands; otherwise, skip to the corresponding $ELSE or $ENDC.
$IFNNULL @n	specifies that if the nth parameter exists, continue executing commands; otherwise, skip to the corresponding $ELSE or $ENDC.
$IFNULL @n	specifies that if the nth parameter does not exist, continue executing commands; otherwise, skip to the corresponding $ELSE or $ENDC.
$IFNX fd	specifies that if fd does not exist, continue executing commands; otherwise, skip to the corresponding $ELSE or $ENDC.
$IFX fd	specifies that if fd exists, continue executing commands; otherwise, skip to the corresponding $ELSE or $ENDC.
$JOB	starts a job, resets end of task code.
$NOCOPY	prevents a listing.
$PAUSE	suspends a CSS file.
$SKIP	skips to $TERMJOB.
$TERMJOB	specifies end of job (EOJ). Any $SKIP or error condition within this job causes CSS processing to transfer to this command with end of task code=255; otherwise, end of task code is defined by the job itself.
$TRANSFER	transfers out of current CSS and enters a new CSS.
$WAIT n	suspends a CSS for a specified number of seconds.
$WRITE	logs a message to the system console and, if present, the system log file.

5.12 COMMAND SUBSTITUTION SYSTEM (CSS) ERROR CONDITIONS

The following is a list of errors possible after issuing CSS commands:

BUFF-ERR
indicates that expanded command line exceeds CSS buffer; skips to $TERMJOB.

FD-ERR
indicates there is not enough space to build a file or required file support not in system; skips to $TERMJOB.

FORM-ERR
indicates command syntax error; skips to $TERMJOB.

IO-ERR
indicates $TERMJOB was found while skipping to $ENDC within a job. Sets end of task code to 255 and ends job. (This is only detected if the conditional that caused the skip was also inside the job; i.e., a skip to $ENDC can skip over a complete job.)

or

EOF was found while skipping to $ENDC; skips to $TERMJOB.

or

EOF was found before ($)ENDB while ($)BUILD a file; skips to $TERMJOB.

JOBS-ERR
indicates second $JOB was encountered; returns control to console.

LVL-ERR
indicates that required CSS levels exceed the number of levels sysgened; returns control to console.

MNEM-ERR
indicates command is not recognized; skips to $TERMJOB.

PAUS-ERR
indicates $PAUSE was entered from system console.

PARM-ERR
indicates command syntax error; skips to $TERMJOB.

SEQ-ERR
indicates a task was active; returns control to console.

TASK-ERR   indicates that currently selected CSS task is
                  not in system; skips to $TERMJOB.

WAIT-ERR   indicates $WAIT was entered from system
                  console.

NOTE

The program skips to $TERMJOB if an error
is detected within a CSS job. The job is
aborted and the next command obeyed is
the first command after the $TERMJOB, at
which point the end of task code is 255.
If the error occurs outside a job,
control is returned to the console.

CHAPTER 6
ACCOUNTING DATA COLLECTION AND REPORTING

6.1 INTRODUCTION

Data regarding system usage is collected via the accounting facility and reported via the accounting reporting utility. The accounting facility is available for systems running in either a multi-terminal monitor (MTM) environment or a non-MTM environment.

6.2 ESTABLISHING THE ACCOUNTING FACILITY IN A MULTI-TERMINAL MONITOR (MTM) ENVIRONMENT

See the OS/32 Multi-Terminal Monitor (MTM) Reference Manual for the procedures involved in establishing the data collection component of the accounting facility. See the System Support Utilities Reference Manual for the procedures involved in reporting accounting facility data.

6.3 ESTABLISHING THE ACCOUNTING FACILITY IN A NON - MTM ENVIRONMENT

Systems operating in a non-MTM environment can also collect and report accounting data through the use of a dummy accounting task.

If a system does not use MTM, an operating system sysgen with accounting support must be performed. A dummy task (MTM taskid), that performs the data collection duties of the accounting facility, is then loaded and started using the following command sequence:

```
LOAD  .MTM/,AFDCP
TASK  .MTM
START , ATF = fd
```

The AFDCP program is able to accept two MTM commands.

```
.MTM QUIESCE
.MTM ATF fd
```

The MTM QUIESCE command terminates AFDCP. The MTM ATF command changes the accounting transaction file (ATF) to the specified file descriptor (fd). The current ATF is closed and a new ATF is allocated (if necessary) and assigned.

Refer to the OS/32 Multi-Terminal Monitor (MTM) Reference Manual for a description of the various accounting facility commands and to the OS/32 System Support Utilities Reference Manual for a description of the Reporting Utility commands.

ALLOCATE fd,
$$
\begin{Bmatrix}
\text{CONTIGUOUS,fsize} \left[, \begin{Bmatrix} \text{keys} \\ 0000 \end{Bmatrix} \right] \\[6pt]
\text{EC} \left[\left[/ \begin{Bmatrix} \text{bsize} \\ 64 \end{Bmatrix} \right] \left[/ \begin{Bmatrix} \text{isize} \\ 3 \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{keys} \\ 0000 \end{Bmatrix} \right] \right] \\[6pt]
\text{INDEX} \left[, \begin{Bmatrix} \text{lrecl} \\ 126 \end{Bmatrix} \right] \left[/ \begin{Bmatrix} \text{bsize} \\ 1 \end{Bmatrix} \right] \left[/ \begin{Bmatrix} \text{isize} \\ 1 \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{keys} \\ 0000 \end{Bmatrix} \right] \\[6pt]
\text{NB} \left[, \begin{Bmatrix} \text{lrecl} \\ 126 \end{Bmatrix} \right] \left[/ \begin{Bmatrix} \text{bsize} \\ 64 \end{Bmatrix} \right] \left[/ \begin{Bmatrix} \text{isize} \\ 3 \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{keys} \\ 0000 \end{Bmatrix} \right] \\[6pt]
\text{ITAM} \left[, \begin{Bmatrix} \text{lrecl} \\ 80 \end{Bmatrix} \right] \left[/ \begin{Bmatrix} \text{bsize} \\ 1 \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{keys} \\ 0000 \end{Bmatrix} \right]
\end{Bmatrix}
$$

APU
$$
\begin{Bmatrix}
\text{LPU} \\[4pt]
\text{n} \left[, \begin{Bmatrix} \text{START} \\ \text{STOP} \\ \text{ON} \\ \text{OFF} \\ \text{ENABLE} \\ \text{DISABLE} \\ \text{LPU=n/ADD} \\ \text{LPU=n/DEL} \\ \text{LPU=*/DEL} \end{Bmatrix} \right]
\end{Bmatrix}
$$

ASSIGN lu,fd
$$
\left[, \left[\begin{Bmatrix} \text{access privileges} \\ \begin{matrix} \text{SRW} \\ \text{SREW} \\ \text{SRO} \end{matrix} \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{keys} \\ 0000 \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{SVC15} \\ \text{SVCF} \\ \text{VFC} \\ \text{HI} \\ \text{LOW} \\ \text{MEDIUM} \end{Bmatrix} \right] \right]
$$

ATTN $\left[\begin{Bmatrix} \text{n} \\ 2 \end{Bmatrix} \right]$

BFILE fd   Used for magnetic tapes and cassettes only

BFILE fd [,lu]   Used for disk devices only

$$\text{BIAS} \left\{ \begin{array}{c} \text{address} \\ * \\ *\text{taskid} \end{array} \right\}$$

BRECORD fd   Used for magnetic tapes and cassettes only

BRECORD fd [,lu] Used for disk devices only

| BUILD fd [,APPEND]
    .
    .
    .
ENDB

CANCEL taskid

$$\text{CLOSE} \left\{ \begin{array}{c} \text{lu}_1 \; [,\text{lu}_2, \ldots, \text{lu}_n] \\ \text{ALL} \end{array} \right\}$$

CONTINUE [address]

DELETE fd_1 [,$\text{fd}_2, \ldots, \text{fd}_n$]

$$\text{DISPLAY ACCOUNTING} \left[, \left\{ \begin{array}{c} \text{taskid} \\ \text{all tasks} \end{array} \right\} \right] \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{system console} \end{array} \right\} \right]$$

$$\text{DISPLAY DEVICES} \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{system console} \end{array} \right\} \right]$$

$$\text{DISPLAY DFLOAT} \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{system console} \end{array} \right\} \right]$$

$$\text{DISPLAY ERRORS} \left[, \left\{ \begin{array}{c} \text{fd} \\ \text{system console} \end{array} \right\} \right]$$

DISPLAY FILES $\left[, \left[\begin{Bmatrix} \text{voln:} \\ \text{default sys vol} \end{Bmatrix} \right] \right] \left[\begin{Bmatrix} \text{filename} \\ * \\ - \end{Bmatrix} \right] \left[. \left[\begin{Bmatrix} \text{ext} \\ - \end{Bmatrix} \right] \right]$

$\left[\begin{Bmatrix} \text{actno} \\ - \\ O \\ S \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{fd} \\ \text{sys console} \end{Bmatrix} \right]$

DISPLAY FLOAT $\left[, \begin{Bmatrix} \text{fd} \\ \text{system console} \end{Bmatrix} \right]$

DISPLAY ITAMTERM ,voln: $\left[\begin{Bmatrix} \text{filename} \\ - \end{Bmatrix} \right] \left[. \left[\begin{Bmatrix} \text{ext} \\ - \end{Bmatrix} \right] \right]$

$\left[, \begin{Bmatrix} \text{fd} \\ \text{sys console} \end{Bmatrix} \right]$

DISPLAY LU $\left[, \begin{Bmatrix} \text{fd} \\ \text{sys console} \end{Bmatrix} \right]$

DISPLAY MAP $\left[/ \begin{Bmatrix} \text{id} \\ \text{.TSK} \\ \text{.PUR} \\ \text{.SEG} \\ \text{.SYS} \\ \text{.OFF} \\ \text{.PST} \\ \text{local memory} \end{Bmatrix} \right] [/\text{rname}] \left[, \begin{Bmatrix} \text{fd} \\ \text{sys console} \end{Bmatrix} \right]$

DISPLAY PARAMETERS $\left[, \begin{Bmatrix} \text{fd} \\ \text{sys console} \end{Bmatrix} \right]$

DISPLAY REGISTERS [,fd]

DISPLAY TASKS [task-id] $\left[, \left\{ \begin{array}{c} \text{fd} \\ \text{sys console} \end{array} \right\} \right]$

DISPLAY TIME $\left[, \left\{ \begin{array}{c} \text{fd} \\ \text{sys console} \end{array} \right\} \right]$

DISPLAY VOLUME ,voln $\left[, \left\{ \begin{array}{c} \text{fd} \\ \text{system console} \end{array} \right\} \right]$

ERROR LOG , $\left\{ \begin{array}{l} \text{ON} \quad \left[, [\text{fd}] \; [, \text{INIT}] \right] \\ \text{OFF} \end{array} \right\}$

ERROR PERIOD $\left[, \left\{ \begin{array}{c} \text{minutes} \\ * \end{array} \right\} \right]$

ERROR RECORDING ,fd, $\left\{ \begin{array}{l} \text{ON} \\ \text{OFF} \end{array} \right\}$

EXAMINE address$_1$ $\left[\left\{ \begin{array}{l} ,n \\ /\text{address}_2 \\ ,1 \end{array} \right\} \right]$ $\left[, \left\{ \begin{array}{c} \text{fd} \\ \text{sys console} \end{array} \right\} \right]$

FFILE fd   Used for magnetic tapes and cassettes only

FFILE fd [, lu]   Used for disk devices only

FRECORD fd   Used for magnetic tapes and cassettes only

FRECORD fd [, lu]   Used for disk devices only

INIT fd $\left[, \left\{ \begin{array}{c} \text{segsize increment} \\ 1 \end{array} \right\} \right]$

IRBUFFER $\left\{ \begin{array}{l} \text{n} \\ \text{display} \\ \text{free} \end{array} \right\}$

$$\text{LOAD} \begin{Bmatrix} \text{fd} [\text{,segsize increment}] \\ \text{taskid} [\text{,} [\text{fd}] [\text{,segsize increment}]] \\ \text{sysid,fd} [\text{,segsize increment}] \\ \text{.BG,fd} [\text{,segsize increment}] \\ \text{.TCM,fd} \\ \text{.LIB,fd} \\ \text{.SEG,fd} \end{Bmatrix}$$

$$\text{MARK dev:,} \begin{Bmatrix} \text{PARAMETER } [\text{,RECL=recl}] \ [\text{,SIZE=size}] \\ \text{OFF} \\ \text{ON} \left[, \begin{Bmatrix} \text{PROTECTED} \\ \text{SYSTEM} [\text{/max acct}] \\ \text{RESTRICTED} [\text{/max acct}] \text{=actno} \end{Bmatrix} \right] \ \left[\text{,CDIRECTORY} \left[= \begin{Bmatrix} \text{bsize} \\ 80 \end{Bmatrix}_{\text{ALL}} \right] \left[/ \begin{Bmatrix} \text{exp} \\ 100 \end{Bmatrix} \right] \right] \end{Bmatrix}$$

$$\text{MEMORY} \begin{Bmatrix} \text{OFF,address} [\text{,size}] \\ \text{ON,address} [\text{,size}] \\ \text{TEST,address} [\text{,size}] \\ n \end{Bmatrix}$$

$$\text{MODIFY address} \left[, \begin{Bmatrix} \text{data}_1 \\ 0 \end{Bmatrix} \right] [\text{,data}_2, \ldots, \text{data}_n]$$

$$\text{OPTIONS} \left[\begin{Bmatrix} \text{AFCONT} \\ \text{AFPAUSE} \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{RESIDENT} \\ \text{NONRESIDENT} \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{SVCCONTINUE} \\ \text{SVCPAUSE} \end{Bmatrix} \right]$$

$$\left[, \begin{Bmatrix} \text{NROLL} \\ \text{ROLL} \end{Bmatrix} \right] \left[, \begin{Bmatrix} \text{LPU } [\text{=n}] \\ \text{NLPU} \end{Bmatrix} \right]$$

PAUSE

REMOVE .SEG,segment name

RENAME oldfd,newfd

REPROTECT fd,new keys

REWIND fd   Used for magnetic tapes and
   or   cassettes only
RW fd

REWIND fd [,lu]   Used for disk devices only
   or
RW fd [,lu]

RVOLUME voln

SEND message [;]

SET LOG [fd] [,COPY] $\left[, \begin{Bmatrix} n \\ 15 \end{Bmatrix}\right]$

SET PRIORITY n

SET SLICE n

SET SYS n

SET TIME $\begin{Bmatrix} [mm/dd/yy],hh:nn:ss \\ OFF \\ 0 \end{Bmatrix}$

SPOOLFILE lu&lul,pseud dev,FORM=formname $\left[, \begin{Bmatrix} VFC \\ IMAGE \end{Bmatrix}\right]$

$\left[, \begin{Bmatrix} NOIMAGE \\ NOVFC \end{Bmatrix}\right]$ $\left[, \begin{Bmatrix} CHECKPOINT \\ NOCHECKPOINT \end{Bmatrix}\right]$ [,COPIES=n] $\left[, \begin{Bmatrix} HOLD \\ RELEASE \end{Bmatrix}\right]$

[,BLOCK= blocksize/indexsize] $\left[, \begin{Bmatrix} DELETE \\ NODELETE \end{Bmatrix}\right]$ [,PRIORITY=p]

START $\left[\begin{Bmatrix} address \\ transfer \ address \end{Bmatrix}\right]$ [,parameter₁,...,parameterₙ]

TASK [taskid]

TCOM segment name,segment size

$$\text{TEMPFILE lu,} \begin{cases} \text{CONTIGUOUS,fsize} \\ \text{EC}\left[\left[/\begin{Bmatrix}\text{bsize}\\64\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{isize}\\3\end{Bmatrix}\right]\right] \\ \text{INDEX}\left[,\begin{Bmatrix}\text{lrecl}\\126\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{bsize}\\1\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{isize}\\1\end{Bmatrix}\right] \\ \text{NB}\left[,\begin{Bmatrix}\text{lrecl}\\126\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{bsize}\\64\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{isize}\\3\end{Bmatrix}\right] \end{cases}$$

$$\text{VOLUME}\left[\text{voln}\left[/\begin{cases}\text{SYSTEM}\\\text{TASK}\\\text{TEMPORARY}\\\text{SPL}\\\text{ROLL}\end{cases}\right]\right]\ \ \text{current voln names}$$

WFILE fd   Used for magnetic tapes and cassettes only

WFILE fd [,lu]   Used for disk devices only

$$\text{XALLOCATE fd,}\begin{cases}\text{CONTIGUOUS,fsize}\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right] \\ \text{EC}\left[\left[/\begin{Bmatrix}\text{bsize}\\64\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{isize}\\3\end{Bmatrix}\right]\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right]\right] \\ \text{INDEX}\left[,\begin{Bmatrix}\text{lrecl}\\126\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{bsize}\\1\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{isize}\\1\end{Bmatrix}\right]\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right] \\ \text{NB}\left[,\begin{Bmatrix}\text{lrecl}\\126\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{bsize}\\64\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{isize}\\3\end{Bmatrix}\right]\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right] \\ \text{ITAM}\left[,\begin{Bmatrix}\text{lrecl}\\80\end{Bmatrix}\right]\left[/\begin{Bmatrix}\text{bsize}\\1\end{Bmatrix}\right]\left[,\begin{Bmatrix}\text{keys}\\0000\end{Bmatrix}\right]\end{cases}$$

XDELETE fd$_1$ [,fd$_2$,....,fd$_n$]

BUILD fd [,APPEND]	$IFE n
ENDB	$IFG n
SET CODE n	$IFL n
$BUILD fd [,APPEND]	$IFNE n
$CLEAR	$IFNG n
$CONTINUE	$IFNL n
$COPY	$IFNNULL @n
$ELSE	$IFNULL @n
$ENDB	$IFNX fd
$ENDC	$IFX fd
$EXIT	$JOB
$NOCOPY	$TRANSFER
$PAUSE	$WAIT $\left[\left\{ \begin{matrix} n \\ 1 \end{matrix} \right\} \right]$
$SKIP	$WRITE
$TERMJOB	

APPENDIX B
OPERATOR COMMAND MESSAGE SUMMARY

ALLO-ERR TYPE=NAME

   Desired filename currently exists on the specified volume.

ALLO-ERR TYPE=SIZE

   Insufficient room available on the disk to allocate the file.

   Block size of indexed file exceeds limit established at sysgen.

   For an indexed file, a zero logical record length or block size was specified.

ALLO-ERR TYPE=TYPE

   The volume specified is not a direct access device. Ensure that specified volume is the disk volume name, not its device name.

ALLO-ERR TYPE=VOL

   Volume name specified or the default volume is not the name of any disk currently online. Ensure that desired disk is online.

APU-ERR |

   Indicates that the task is auxiliary processing unit (APU) |
   active and the APU is being controlled by another task, or |
   that the command processor was unable to gain access to the |
   APU. |

ARGS-ERR

   The amount of space between CTOP and UTOP is insufficient for the command processor to place the parameters of the START command.

ASGN-ERR

Optional file descriptor (fd) or disk device of voln could not be assigned; e.g., fd or disk is assigned for exclusive use only.

ASGN-ERR TYPE=BUFF

An attempt was made to assign a file when there was insufficient system space available to accommodate the file control block (FCB). Close any currently assigned files that are no longer required, or increase the size of system space with a SET SYS command.

ASGN-ERR TYPE=LU

An attempt was made to assign a logical unit (lu) that is greater than the maxlu number specified at Link time.

ASGN-ERR TYPE=NAME

An assignment was directed to a nonexistent file.

ASGN-ERR TYPE=PRIV

A file that is currently assigned to an lu with a given privilege could not be assigned to another lu because the access privileges were in conflict. Request a compatible access privilege on second assignment or change the access privileges currently associated with the file.

ASGN-ERR TYPE=PROT

The file assigned is unconditionally protected or the read/write keys specified in the assign statement do not correspond to those associated with the file.

ASGN-ERR TYPE=SIZE

An indexed file was assigned and there is not enough room on the disk to allocate a physical block. Ensure sufficient space on the disk by deleting old files or reducing the block size of the file. Alternatively compress the disk using the Disk Backup Utility.

ASGN-ERR TYPE=SPAC

An attempt was made to assign a file that required buffer space exceeding the task maximum system space allotment.

ASGN-ERR TYPE=TGD

> An attempt was made to assign a trap generating device that does not support such assignment.

ASGN-ERR TYPE=VOL

> Volume name specified or default volume is not the name of any disk currently online.

ATTN-ERR

> The priority specified in an ATTN command was not a decimal number from 2 through 249.

fd-BAD LINE COUNT

> The value of n in a .SPL FORWARD or BACKWARD command exceeded file limits.

BPAC-ERR

> The disk was not ready or was not readable. If the disk is ready and is not write-protected, reformat using the disk initializer utility.

BUFF-ERR

> The expanded command substitution system (CSS) line exceeds the CSS buffer size. Specify larger CSS buffer length at next sysgen, or modify CSS statements. Ensure that the expanded CSS line does not overflow the buffer by shortening the length of the unexpanded line.

CLOS-ERR TYPE=LU

> The lu number is greater than maxlu specified at Link time.

CLOS-ERR TYPE=BUFF

> System space has become corrupted and buffers and/or FCBs cannot be returned to the free system space.

fd IS NOT A CONTIGUOUS FILE

> The fd specified in an INIT command is not a contiguous file.

DATE-ERR

An ERROR LOG command was entered with the ON parameter specified, but the date and time parameters of the SET TIME command were not specified.

DELE-ERR TYPE=ASGN

An attempt was made to delete a file that is currently assigned.

DELE-ERR TYPE=BUFF

There is insufficient memory available in system space to perform the delete operation. Enlarge system space with a SET SYS command or make more space available by closing one or more logical units.

| DELE-ERR TYPE=NAME

| The file specified does not exist.

| DELE-ERR TYPE=PRIV

| The file is currently assigned to a task.

| DELE-ERR TYPE=PROT

| The read/write protection keys are not 0.

DELE-ERR TYPE=TYPE

The volume name specified or default volume is not a direct access device.

| DELE-ERR TYPE=VOL

| The specified volume is not mounted, the volume was not
| marked on, or the incorrect volume name was given.

DEV-ERR

An attempt was made to mark on or off a nonexistent device or a pseudo device; or an attempt was made to modify the attributes of a nonexistent device or bulk storage device.

DEVICE NOT A DISK

The specified device with voln is not a bulk device.

DIR-ERR

A DISPLAY FILES command was directed at a disk containing one
or more invalid directory entries. Run the disk integrity
check utility to verify the contents of the disk pack.

DUPL-ERR

When marking on a direct access device, the volume name
associated with the device was an existing device or volume
name. Run the disk initializer utility to change the disk
volume name, or mark the other disk offline.

ERRC-ERR

A DISPLAY ERRORS or ERROR RECORDING command was entered, but
error recording was not specified at sysgen.

xxxx ERROR ON fd SECTOR n

An input/output (I/O) error occurred when an attempt was made
to initialize sector n of file fd.   xxxx is the type of
error.

FD-ERR

The fd was syntactically incorrect; or a program on the disk
was being loaded, and there was not enough system space for
the load operation.

FETCH ATTR ERR

A fetch attributes failed on secondary directory when marking
on protected with secondary directory.

FILE NOT FOUND

The specified file was not found.

FILE filename NOT ON QUEUE

The file specified by the PURGE parameter of the .SPL command
is not on the Spooler queue.

FILE filename SPOOLING

The file specified by the PURGE parameter of the .SPL command
is currently being spooled out.

FORM-ERR

The command format was syntactically incorrect.

IDLE-ERR

A .SPL REWIND, CANCEL, FORWARD, BACKWARD, or CONTINUE command
was isssued to an idle device. Verify the name of the device
to which the command is directed.

INVALID FD=fd

The filename specified is a device, not a file, or the device
specified is not a valid pseudo device.

INVALID SYNTAX fd

An invalid fd was specified with the PRINT or PUNCH parameter
of the .SPL command.

I/O-ERR

A device accessed by the command processor returned a nonzero
I/O status. These type fields can be displayed:

TYPE=PRTY - Parity or other recoverable error. Retry the
operation with another unit, if possible.

TYPE=UNRV - An unrecoverable error occurred.

TYPE=UNRV SEGMENT=SS TYPE=IMPU - An error occurred while
writing the impure segment of the segment name (SS) out to
its roll file in order to read task R3 into memory. No
action is necessary. The segment SS has been flagged
nonrollable.

TYPE=EOF,EOM - The device reached EOF or EOM before
completing the operation.

TYPE=DU - The device is unavailable. Ensure that the device
is online and ready.

TYPE=FUNC - An invalid operation is being directed toward a
device; e.g., attempting to write to a read-only device.

TYPE=LU - Illegal or unassigned lu. Close and reassign
proper lu.

INIT-ERR

Memory error recording was initialized, but it already had been specified and was in progress.

IRBUF - ILLEGAL PARAMETER

The number of buffers requested by IRB n exceeds the maximum 99 buffers.

The command IRB was entered with no parameter.

IRBUF - n BUFFER(S) IN USE

Fewer buffers were requested than were currently in the queue.

IRBUF - BUFFERS CAN'T GET SYSTEM SPACE

The requested number of buffers specified by IRB n exceeds the amount of available system space.

IRBUF - nn BUFFER(S) IN USE

The IRBUF command was entered with the "free" option, and the current buffers are active.

IRBUF - nn BUFFER(S) FREED

The IRBUF command was entered with the "free" option, and all buffers are inactive.

JOBS-ERR

A $JOB statement was encountered following another $JOB statement but prior to a $TERMJOB statement.

LOAD-ERR TYPE=IO

An I/O error was generated during the load operation. Retry the load operation. If the same condition results, verify the status of the medium from which the task is being loaded.

LOAD-ERR TYPE=LIB

The data in the LIB was invalid. This error most frequently occurs when an attempt is made to load a task which was not established with Link.

LOAD-ERR TYPE=LOPT

Conflict between load options requested and those specified at Link time.

LOAD-ERR TYPE=MAP

While loading a task, an attempt was made to automatically load a sharable segment, but fd could not be found or the shared segment table overflowed.

LOAD-ERR TYPE=MEM

A load was attempted when no memory area large enough was available. Change priorities or rollability of current tasks to allow a roll operation to occur, or cancel one or more current tasks.

LOAD-ERR TYPE=MTCB

An attempt was made to load more tasks than the system permits.

LOAD-ERR TYPE=NOFP

An attempt was made to load a task requiring floating point support, and the required floating point option is not supported in the system.

LOAD-ERR TYPE=PRES

The specified taskid is already present in the system.

LOAD-ERR TYPE=PURE

Duplicate pure segment name.

LOAD-ERR TYPE=ROIO

I/O error encountered in writing roll file. Retry the load operation. If the same condition occurs, verify the status of the roll volume.

LOAD-ERR TYPE=RVOL

Allocation or assignment error on roll file. Ensure that the roll volume is online and write enabled. Retry the load operation.

LOAD-ERR TYPE=SEG

An attempt was made to load a task requiring the run-time library (RTL) or a TCOM prior to establishing an RTL or TCOM segment.

LOAD-ERR TYPE=SPAC

Not enough system space for loader task control block (TCB). Increase system space.

LU-ERR

An invalid lu number or lu assignment was attempted; an invalid fd was encountered; or a nonzero account number was specified.

LVL-ERR

The number of sysgened CSS nesting levels was exceeded.

MAP-ERR

The logical processing unit (LPU) is already mapped to an APU.

MEM-ERR

The system contains insufficient memory to support the command request.

MEM-ERR TYPE=ADDRESS

Indicates that address is outside task or system space.

MEM-ERR TYPE=FIND

Indicates that requested system space is not free or requested memory is located in another segment.

MEM-ERR TYPE=NOMD

Indicates that the MEMORY command was entered, but the system does not support memory diagnostics.

| MEM-ERR TYPE=SIZE POS=xxx

    Indicates that calculated end address is between task and
system space or is outside system space. xxx is the
requested value.

| MEM-ERR TYPE=SPAC

    Indicates that insufficient system space exists to mark off
the specified memory area; memory is not marked off.

MNEM-ERR

    The entered command was not recognized.

NOAC-ERR

| Indicates that accounting is not supported for this system.

| NO ACTIVE TASK(S) FOUND

| Indicates no active tasks found in the system.

NOBC-ERR

    A BFILE, BRECORD, FFILE, or FRECORD command was entered and
bulk file command support is not included in the operating
system.

NODA-ERR

    Direct access support was not included in operating system.

NO DIRECTORY ENTRIES ON voln

    The specified volume has no files on it.

NOFF-ERR

    An attempt was made to mark on a disk device when the
integrity of the data on the disk was questionable. Run the
Disk Integrity Check Utility on the disc. The disk can be
marked on protected.

NOFP-ERR

    No floating point support exists in the system.

NOPR-ERR

A command was entered that required more parameters than specified in the command line.

NROL-ERR

An OPTIONS ROLL command was directed to a nonrollable task.

NO SYSTEM SPACE

An attempt to mark a disk on with a secondary directory option failed because there was not enough system space available for creating a secondary directory block.

NSUP-ERR

The device does not support error recording.

NULL-ERR

An attempt was made to rename the null device.

OFF-ERR

An ERROR LOG command specified the OFF parameter, but the error recording function was already off.

ON-ERR

An ERROR LOG command was entered twice with the ON parameter specified.

OPT-ERR

A conflict exists between requested options and options specified at Link time.

OSP-ERR

A device specified in a .SPL REWIND, CONTINUE, FORWARD, or BACKWARD command is not an output spool device.

PARM-ERR

A command was entered with invalid parameters.

PERD-ERR

The number of minutes specified for the error log readout period was not a number from 1 through 1440.

PRES-ERR

An ERROR LOG command was entered, but the operating system does not support error reporting.

PRI-DIR READ ERR

An attempt to mark a disk on with a secondary directory option failed because an I/O error occurred when reading the primary directory.

PRIV-ERR

The access privilege mnemonic was syntactically incorrect.

PRTY-ERR

A SET PRIORITY command was entered, and the requested priority is greater than the maximum priority set at Link time.

READ-ERR

An attempt was made to mark a disk on that is not hardware enabled for write, or that returns an I/O error when it is accessed.

| REM-ERR TYPE=NAME

| Indicates that shared segment name does not exist.

| REM-ERR TYPE=RMV

| Indicates that task common to be removed is a global task
| common that is not removable.

| REM-ERR TYPE=USE

| Indicates that shared segment is presently in use.

RENM-ERR TYPE=ASGN

The file or device cannot be assigned for ERW (required to perform the rename) because the file or device is currently assigned to at least one lu.

RENM-ERR TYPE=BUFF

Indicates that an error occurred when closing an lu for a RENAME, or the system space control blocks are corrupted.

RENM-ERR TYPE=NAME

The new filename already exists in the volume directory, or the new device name already exists within the DMT.

REPR-ERR TYPE=ASGN

The file or device cannot be assigned for ERW (required for reprotection) because the file or device is currently assigned to at least one lu.

ROLL-ERR

A BIAS * command was entered and the currently selected task is rollable, or BIAS * taskid was entered and taskid is a rollable task.

SEC-DIR ALLO ERR

An attempt to mark a disk on with a secondary directory option failed because there was insufficient space on the disk or the disk was write-protected (hardware feature).

SEC-DIR ASGN ERR

Assignment to secondary directory failed during marking on protected with secondary directory.

SEC-DIR DELE ERR

DELETE command to old secondary directory failed because old file was not properly closed.

SEC-DIR NOT PRESENT

Secondary directory does not exist on disk and mark on protected with secondary directory was attempted.

SEC-DIR READ ERR

An attempt to mark a disk on protected with a secondary directory option failed because an I/O error occurred when reading the secondary directory.

SEC-DIR VERIFY ERR

Secondary directory failed to verify with the primary directory when marking on protected with secondary directory.

SEC-DIR WRIT ERR

An attempt to mark a disk on with a secondary directory option failed because an I/O error occurred when establishing the secondary directory on the disc.

SEGMENT REQUEST NOT FOUND

A DISPLAY MAP command was entered, and the requested segment mask was not found, or no tasks exist in memory.

SEQ-ERR

A command was entered out of sequence when:

● attempting to pause a task when none was active,

● assigning of a currently assigned lu while a task was active, or

● entering an OPTION command for an active task.

SKIP-ERR

An attempt was made to skip beyond the end of a CSS job. The CSS job concept delimits CSS jobs with the $JOB and $TERMJOB statements. Conditional CSS statements allow skipping to a $ENDC if certain conditions are met. If the nesting of conditional statements is incorrect, a $TERMJOB statement can be encountered prior to terminating all of the conditional statements.

SLOC-ERR

The starting location of a task was specified below UBOT or was omitted when it was required.

SPAC-ERR

An assign on behalf of a task was refused because system
space available for task use was exceeded. Reestablish the
task with a larger maximum system space.

SPECIFIED TERMINAL(S) NON-EXISTENT FOR voln

The communications terminal specified in a DISPLAY ITAMTERM
command was not found, or voln has no LCBs allocated for it.

STAT-ERR

An attempt was made to mark a device on or off while an lu
was assigned to it.

SVC6-ERR TYPE=ARGS

Insufficient memory exists between UTOP and CTOP to pass all
parameters.

SVC6-ERR TYPE=DORM

An attempt was made to issue an SVC 6 to a task that was in
the dormant state.

SVC6-ERR TYPE=NMSG

A SEND command was entered and the receiving task could not
receive a message trap.

SVC6-ERR TYPE=PRES

An attempt was made to send a message to a nonexistent task.

TASK-ERR

A task related command was entered and there was no currently
selected task.

TASK(S) NOT FOUND

The specified task was not found in the system; no tasks
found in the system.

TCOM-ERR

A TCOM command failed for one of the following reasons:

o TYPE=MEM – No vacant memory area of sufficient size exists.

o TYPE=NAME – A task common of the same name already exists.

o TYPE=SYS – Not enough system space exists for required segment descriptor entry.

TKID-ERR

An invalid taskid syntax was entered on a LOAD command.

VOLN-ERR

Indicates that the volume is not online or the volume name is invalid.

WRIT-ERR

An attempt was made to mark on a device that is hardware protected without the PROTECT option. Use the PROTECT option in the MARK command.

taskid: ACCESS PRIVILEGE ADDRESS ERROR AT RRxxxx (yyyyyy)

The user-program tried to perform a function (execute, store, or load) which is prevented by the access privileges requested at Link time for the segment. The most common cause of this error is an attempt to store data into a pure segment. Program address is RRxxxx; segmentation register is RR; physical address is yyyyyy.

taskid: ADDRESS FAULT IN SVC AT xxxxx (yyyyyy)

Address of supervisor call (SVC) parameter block or an address parameter in the parameter block points to a data structure which is outside the task taskid memory allocation, or does not point to a data structure which is properly aligned.

taskid: ALIGNMENT FAULT INSTRUCTION AT xxxxxx (yyyyyy)
          MEMORY FAULT ADDRESS=xxxxxx (yyyyyy)

Data instruction not properly aligned to specific fields for fullword or halfword alignment. The memory fault address is the memory location that is not properly aligned. The memory fault address is given only on Perkin-Elmer series 3200 machines. Program address is xxxxxx; physical address is yyyyyy.

taskid: ARITHMETIC FAULT AT xxxxx (yyyyyy)

Arithmetic fault is detected at location xxxxx in the taskid address space (physical address yyyyyy).

taskid: END OF TASK n

Task taskid has ended. The end of task code in decimal is n.

taskid: ILLEGAL INSTRUCTION AT xxxxx (yyyyyy)

Illegal instruction fault detected at location xxxxx in the taskid address space (physical address yyyyyy).

taskid: ILLEGAL SVC AT xxxxx (yyyyyy)

    Illegal SVC call at location xxxxx in the taskid address
space (physical address yyyyyy).

taskid: INVALID SEGMENT ADDRESS ERROR AT xxxxx (yyyyy)

    The task tried to address a segment outside the address space
of the program. Program address is xxxxx; physical address
is yyyyy.

taskid: I/O-ERR TYPE=xxxx SEGNAME=yyyy SEGTYPE=2222

    Input/output (I/O) error type xxxx was encountered while
trying to roll out the segment named yyyy to the roll volume.
Task space is needed to load or roll in task taskid. Segment
yyyy is set as nonrollable, and write error flag is set in
the segment descriptor entry (SDE). To insure full integrity
of the segment, reload the pertinent module(s).

taskid: MEMORY PARITY ERROR AT xxxxx (yyyyyy)

    Parity or an error correction code (ECC) machine malfunction
is detected at location xxxxx (physical address yyyyyy).

taskid: SEGMENT LIMIT ADDRESS ERROR AT RRxxxx (yyyyyy)

    The task attempted to access an address outside allowable
limits for one of its segments. Program address is RRxxxx;
segmentation register is RR; physical address is yyyyyy.

taskid: TASK PAUSED

    Task taskid paused. Results from SVC 2 code 1 or operator
PAUSE command.

taskid>

    SVC 1 read request to console device from task taskid. Data
should be entered as soon as possible to prevent blocking the
console.

SYSTEM RELATED MESSAGES:

CDIR FULL-devn

    The secondary directory on disk devn is full. Refer to the
MARK command description in Chapter 3 for required action.

FLOATING POINT HARDWARE NOT PRESENT

This is a warning message that occurs if the operating system is sysgened with hardware floating point support, but at loading time, the operating system is loaded on a machine that does not support hardware floating point. In this case, the operating system will run, but it cannot successfully run programs requiring floating point support.

I/O ERROR ON voln; MARK OFF AND CHECK
BIT MAP ERROR ON voln; MARK OFF AND CHECK

voln is the disk volume name. An I/O error is reported to the system when reading or writing a directory block, bit map sector, or volume descriptor (map sector 0). A bit map error is reported to the system console in the event of a bit map error; i.e., attempting to allocate an already allocated sector or volume, or attempting to release an already released sector.

When either of these errors occurs, any operations that require bit map changes are rejected with device unavailable status. This action permits read and write of preallocated contiguous files and read only of the preallocated index files. An attempt to allocate or delete a file is rejected with device unavailable status. The operator is advised to run a disk check on that disk.

OS32MTrr-uu

Printed after system initialization. The release level is rr; uu is the update level.

PIC NOT ACTIVE AT ADDRESS XX

When the operating system is generated, the precision internal clock (PIC) address is specified. At operating system START time, the operating system checks to see if the PIC is operational at that address. If not, the above message is displayed. The operating system continues to run, but without the PIC.

POWER RESTORE - RESET PERIPHERALS

Power fail restore sequence; no operator response required.

POWER RESTORE - RESET PERIPHERALS AND ENTER GO

Power fail restore sequence; perform any manual intervention required at the peripheral device(s), then type GO (CR) to complete power recovery.

The following are system related messages for users of the Model 3200MPS Systems:

APB LINKAGE ERROR, "APU-n" IDLE

During auxiliary processing block (APB) queue processing, the specified auxiliary processing unit (APU) detected an APB linkage fault.

Faults indicated include corrupted pointer values, improper address alignment, and queue overflow.

APB QUEUE ACCESS TIME-OUT "APU-n" IDLE

The specified APU was not able to gain exclusive access to either the corresponding APU ready queue or the central processing unit (CPU) receive queue within the allowable time frame (two minutes maximum).

This time-out may occur as a result of heavy activity by various APUs accessing the CPU receive queue; or if, upon recognition by the APU of a task control block (TCB) count transition from zero to nonzero for the APU TCB queue, queue access could not be gained before expiration of the hardware time-out.

This message can result from the following CPU control functions received by the APU over the real-time support module (RTSM).

START EXECUTION
RESCHEDULE
RESCHEDULE FROM APU TO CPU

COMMAND RESPONSE FAILURE, "APU-n" STATE INDETERMINATE

The specified APU does not respond to control functions initiated by the CPU via the RTSM.

HOST/APU CONTROL FUNCTION ERROR, "APU-n" IDLE

The specified APU received an unrecognizable command from the host via the RTSM.

This indicates a control function parity error or the receipt of an unsupported command byte.

LINK CHECK FAILURE, "APU-n" IDLE

A link check sequence failure occurred between the host and the specified APU.

This failure may be detected during the APU power-up sequence or due to the execution of the link check control function initiated by the CPU.

TCB LINKAGE ERROR, "APU-n" IDLE

During TCB queue processing, the specified APU detected a TCB linkage fault.

Faults indicated include corrupted pointer values and improper address alignment.

APPENDIX D
SYSTEM CRASH CODES

A system crash occurs when the operating system detects an irrecoverable internal error. When a crash occurs, the system displays a message on the system console, informing the operator of the system crash and gives a hexadecimal crash code number. This number can be used as a diagnostic to help determine the cause for system failure.

The system source sysgen parameter, SGN.SAFE, controls the inclusion of safety checking code into the operating system modules. This code is dispersed throughout the system. It performs consistency checks at various places in the operating system code. If a consistency check fails, then a system crash occurs.

Generally, operating systems provided to the customer are assembled with SGN.SAFE set to zero because the safety check code increases operating system overhead. However, if modifications are to be made to the system, it is recommended that the system be reassembled with SGN.SAFE equated to 1 to assemble in the checks. These checks allow the operating system to detect internal errors sooner than otherwise possible, making it easier to track down problems.

The system signals a crash by executing a SINT instruction for device zero. The resulting input/output (I/O) interrupt causes control to be passed to the crash handler. The crash handler prints out the crash message, which includes the crash code.

After the system has crashed, data contained in registers 0, 1, 2, and 3 of register set 0 have been destroyed. Registers 0 and 1 of register set 0 contain the program status word (PSW) that was active at the time that the crash SINT instruction was executed. Note that the destruction of data contained in these registers makes it more difficult to track down problems in interrupt service routines (ISRs) because these routines use registers 0 through 7 of register set 0. In the following list of crash codes, an asterisk following the crash code indicates that the crash code can occur only in systems assembled with SGN.SAFE equated to 1. The other crash codes can occur in any system. The crash code descriptions include a description of registers containing information that might help find the problem. The register set referred to in these descriptions is the same register set selected by the PSW saved in register 0 of set 0. Register mnemonics R0 through R9, and RA through RF, correspond to registers 0 through 9, and 10 through 15, of the register set specified.

Crash code: 1
Modules: CMON, CMSP

The system console device mnemonic from SPT.IVT was not found in
the DMT, or CMON was unable to start CMDP. CMDP could not assign
the system console device, or the write of the operating system
identifier message returned bad status.

Crash code: 3
Module: CMON

CMON received an unexpected item on its task queue. R15 contains
the item that was found on the task queue.

Crash code: 4
Module: EXIN

The address of the memory access controller (MAC) is greater than
the value specified in the SYSGEN/32 DEVADS statement.

| Crash code: 5
| Module: EXIN

| An operating system sysgened with hardware floating point support
| attempted to execute floating point instructions on a machine not
| equipped with floating point hardware.

Crash code: 7
Module: CMDB

An online disk was discovered to have no associated entry in the
VMT.

Crash code: 10
Module: CMDB

An attempt to mark a disk on or off occurred, but direct access
support is not included in the system. This crash can occur only
when disk devices are included in a system without direct access
support.

Crash code: 100
Module: EXIN

An arithmetic fault occurred while a user task was executing in
RS or RSA state. The contents of register set 0 at the time the
fault was detected are saved at EREGS. Registers RE and RF
contain the interrupt old PSW.

Crash code: 101
Module: EXIN

An arithmetic fault occurred while executing system code, but RS
or RSA state was not entered on behalf of a user task (u-task).
The contents of register set 0 at the time the fault was detected
is saved at EREGS.   Registers RE and RF contain the interrupt old
PSW.

Crash code: 102
Module: EXIN

An illegal instruction was detected in system code.   The contents
of register set 0 at the time the fault was detected are saved at
EREGS.   Registers RE and RF contain the interrupt old PSW.

Crash code: 106
Module: EXIN

A supervisor call (SVC) interrupt occurred, but the SVC
instruction that caused the interrupt was not found as expected.
RA contains the relocated (real) address of the instruction
following the instruction that caused the interrupt.   RC contains
the unrelocated (virtual) address of the instruction following
the instruction that caused the interrupt.   The contents of
register set 0 at the time the fault was detected are saved at
EREGS.   Registers RE and RF contain the interrupt old PSW.

Crash code: 107*
Module: EXTM

TMUCHN is trying to remove a task with an invalid task control
block (TCB) address from the ready queue.   Register R9 contains
the address of the current TCB.

Crash code: 108*
Module: EXTM

TMREMW is trying to remove wait conditions from a task whose TCB
address is invalid.   Register R9 contains the invalid TCB
address.   Register RD contains the wait bit masks that are to be
removed.

Crash code: 109*
Module: EXTM

TMRDISP is trying to dispatch the task at the top of the ready
queue, but the TCB address of this task is invalid.   Register R9
contains the invalid TCB address.

Crash code: 10A*
Module: EXTM

TMENQPRI discovered a ready queue count of zero, but the queue front pointer was not zero.

Crash code: 10B*
Module: EXTM

TMCHN is trying to put a task with an invalid TCB address onto the ready queue. Register R9 contains the invalid TCB address.

Crash code: 10C*
Module: EXTM

An attempt was made to add an item to a task's task queue, but the TCB address of the task is invalid. Register R9 contains the invalid TCB address. Register RA contains the parameter that is to be added to the task queue.

Crash code: 10D*
Module: EXTM

A task is being dispatched into user state by TMRDISP, and the task owns the user register set, but not the MAC/MAT. Register R9 contains the TCB address of the task. Register RC contains the value that was expected in SPT.MCOW.

Crash code: 10E*
Module: EXTM

TMRDISP found that the last task executing was interrupted during an interruptible instruction. The TCB of this task was obtained in order to save the scratchpad registers, but was found to be invalid. Register RB contains the TCB address of the task being dispatched. Register R9 contains the TCB address of the interrupted task. Registers RE and RF contain the dispatch PSW.

Crash code: 10F*
Module: EXTM

TMENQTL discovered a ready queue count of zero, but the queue front pointer was not zero.

Crash code: 110*
Module: EXTM

TMSTRT is trying to start a dormant task, but the TCB address is
invalid. Register R9 contains the invalid TCB address. Register
RF contains the unrelocated (virtual) starting address of the
task.

Crash code: 111*
Module: EXTM

TMREMW was trying to remove wait conditions from a task, but the
wait conditions to be removed were not valid. Register R9
contains the TCB address. Register RD contains the mask for the
wait bits that are to be reset.   RB contains masks for the
invalid wait bits that caused the crash.

Crash code: 112* |
Module: EXTM |

TMSTART entered in wrong system state (not NS). |

Crash code: 113* |
Module: EXTM |

TMSTART discovered that the user's context area pointer, |
TCB.UCTX, was zero. |

Crash code: 114* |
Module: EXTM |

TMSTOP entered in wrong system state (SQS interrupts not |
disabled). |

Crash code: 115*
Module: EXTM

TMSTOP finds that the TCB address of the task whose registers are
to be saved is invalid. Register R9 contains the address of the
TCB.

Crash code: 116* |
Module: EXTM |

TMREMW entered in wrong system state (SQS interrupts not |
disabled). |

| Crash code: 117*
| Module: EXTM

| TMSETW entered in wrong system state (SQS interrupts not
| disabled).

Crash code: 118
Module: EXTM

TMUCHN is trying to remove from the ready queue a task that is
absent from the queue. Register R9 contains the address of the
TCB.

Crash code: 119
Module: EXIN

A MAC or memory address translator (MAT) fault occurred while
executing system code. The contents of register set 0 at the
time the fault was detected are saved at EREGS. Registers RE and
RF contain the interrupt old PSW.

| Crash code: 11A*
| Module: EXTM

| TMATQ entered in wrong system state (SQS interrupts not
| disabled).

| Crash code: 11B*
| Module: EXTM

| TMSTSW entered in wrong system state (SQS interrupts not
| disabled).

| Crash code: 11C*
| Module: EXTM

| TMDISP entered in wrong system state (not NS).

| Crash code: 11D*
| Module: EXTM

| TMRDISP entered in wrong system state (not NS).

| Crash code: 11E*
| Module: EXTM

| TMCDISP entered in wrong system state (not NS).

Crash code: 11F*
Module: EXTM

TMSETW was passed an invalid TCB address.   Register 9 contains
the invalid TCB address.

Crash code: 120*
Module: EXTM

TMSETW was called to set an invalid task wait condition.
Register 11 contains the invalid wait bits that were specified.

Crash code: 121*
Module: EXMY

RELMEM is trying to release a task memory block not in task
memory; or, during the release of the memory, the free list for
task memory was found not to be in increasing address order.

Crash code: 122*
Module: EXMY

RELSYP is trying to release a system space block that does not
exist in system space; or, during the release of the memory, the
free list for system space was found not to be in decreasing
address order.   Register R9 contains the address of the current
TCB.   The value contained in register R9 can be zero.

Crash code: 131
Module: EXIN

An SVC was issued, but there is no current task.   The contents of
register set 0 at the time the fault was detected are saved at
EREGS.   Registers RE and RF contain the interrupt old PSW.

Crash code: 132
Module: EXIN

An illegal SVC call was issued from system code. This was caused
by issuing an SVC with an invalid SVC number, or the execution of
an SVC instruction in a system state that does not permit SQS
interrupts.   The contents of register set 0 at the time the fault
was detected are saved at EREGS.   Registers RE and RF contain the
interrupt old PSW.

Crash code: 142
Module: EXIN

An SVC has an invalid parameter block address, or a parameter
inside the parameter block is invalid. This could be caused by
not having the parameter block fullword aligned. The contents of
register set 0 at the time the fault was detected are saved at
EREGS. Registers RE and RF contain the interrupt old PSW.

Crash code: 145
Module: EXIN

An invalid reason code was received during a MAT fault. The
contents of register set 0 at the time the fault was detected are
saved at EREGS. Registers RE and RF contain the interrupt old
PSW.

Crash code: 150
Module: FLTP

A bad vector table offset was generated by the floating point
emulator while processing an op-code to see if it was a floating
point instruction.

Crash code: 152
Module: EXIN

A parity or error correction code (ECC) error occurred while
executing system code. This fault can be caused by referencing
an address outside of physical memory or by a noncorrectable
memory error. The contents of register set 0 at the time the
fault was detected are saved at EREGS. Locations X'20' through
X'27' contain the current PSW at the time of the machine
malfunction interrupt. For Perkin-Elmer 3220, 3230, and 3240
machines, locations X'2C' through X'2F' and X'44' through X'47'
contain the physical address that caused the interrupt. The word
at locations X'40' through X'43' contains a reason code for the
interrupt.

Crash code: 153
Module: EXIN

An instruction data format fault or alignment fault occurred in
system code. This fault can only occur on a Perkin-Elmer 3200
Series machine. The contents of register set 0 at the time the
fault was detected is saved at EREGS. Registers RE and RF
contain the interrupt old PSW.

Crash code: 175
Module: EXSP

An SVC 2 call was issued from system code, but the RS register
set is not owned by the caller. Register R9 contains the address
of the current TCB. Register RA contains the value that was
expected to be found in either SPT.RSOW or SPT.UTOW.

Crash code: 176*
Module: EXTM

TMRSRSA or TMRSARS found that the RS register set is not properly
owned by the calling task. Register R2 contains the address of
the current TCB. Register R8 contains the value that was
expected to be found in either SPT.RSOW or SPT.UTOW.

Crash code: 177
Module: EXTM

An interruptible instruction was executed within the system code
and interrupted. This condition is detected when a task is being
dispatched into system state by TMRD.SYS. Register R9 contains
the address of the TCB. Registers RE and RF contain the dispatch
PSW.

Crash code: 178*
Module: EXTM

TMRD.SYS, TMRSIN, or TMRSAIN detected an attempt to enter RS/RSA
state with floating point enabled in the PSW.

Crash code: 180*
Module: EXTI

The LFC event service routine was dispatched even though no
outstanding request existed for dispatch of this routine.

Crash code: 185*
Module: EXTI

TOCHOFF found the head of the DCB timeout chain to be zero. This
should never happen since the chain uses nonzero address pointers
or -1 to signify the end of the chain. Or, TOCHOFF found that
the DCB that it was trying to remove from the DCB timeout chain
was not on the chain. Register RD contains the address of the
DCB that was to be removed from the timeout chain.

Crash code: 186*
Module: EXTI

The PIC or LFC event service routine found that the head of the timer chain that it was servicing was zero. This should not occur since the chain uses either a nonzero address pointer or -1 to signify the end of the chain.

Crash code: 190
Module: EXTM

A task manager queueing routine was unable to gain exclusive access to a task queue before timing out.

Crash code: 191*
Module: EXTM

TMENQPRI entered in wrong system state (SQS interrupts not disabled).

Crash code: 192*
Module: EXTM

TMENQTL entered in wrong system state (SQS interrupts not disabled).

Crash code: 193*
Module: EXTM

TMDEQ entered in wrong system state (SQS interrupts not disabled).

Crash code: 194*
Module: EXTM

TMDEQHD entered in wrong system state (SQS interrupts not disabled).

Crash code: 195*
Module: EXTM

TMRDYENQ entered in wrong system state (SQS interrupts not disabled).

Crash code: 196*
Module: EXTM

TMRDYDEQ entered in wrong system state (SQS interrupts not disabled).

Crash code: 197*
Module: EXTM

TMRINENQ entered in wrong system state (SQS interrupts not disabled).

Crash code: 198*
Module: EXTM

TMRINDEQ entered in wrong system state (SQS interrupts not disabled).

Crash code: 1F0
Module: EXIN

A system queue service interrupt occurred, but the interrupt old PSW status had the system queue service interrupt enable bit turned off. The contents of register set 0 at the time the fault was detected are saved at EREGS. Registers RE and RF contain the interrupt old PSW.

Crash code: 1F1
Module: EXIN

On a machine with two register sets, the user register set was not available for exclusive use. This can be caused by having an event service routine not return via SQSEXIT. The contents of register set 0 at the time the fault was detected are saved at EREGS. Registers RE and RF contain the interrupt old PSW.

Crash code: 203*
Module: EXIO

COMQ found that the queueing number that was passed to it as a parameter was not in the range of 1 to 4. Register R9 contains the queueing parameter that was passed to COMQ. This number should not be in the range of 1 to 4. Register RB contains the leaf to which the IOB is being queued. Register RA contains the IOB that is being queued to the leaf.

Crash code: 220*
Module: EXIO

GETIOB found that no IOBs were available when it tried to allocate one. This should never happen because task is placed into connect wait until an IOB becomes free at the time the last remaining free IOB is allocated. Register R9 contains the address of the TCB.

Crash code: 221*
Module: EXIO

RELIOB is trying to release an IOB that is not found in the TCB
IOB list.   It could be either an IOB from a DCB, or a corrupted
IOB. Register R9 contains the address of the TCB.   Register RB
contains the IOB type.   The crash occurred because this number
was not 1. Register RA contains the address of the IOB that
RELIOB attempted to release.

Crash code: 230
Module: EXIO

A leaf that was added to the system queue had the address of its
event service routine as 0. Register RF contains the address of
the leaf being processed. Register RD contains the DCB address
that was obtained from the leaf.

Crash code: 250
Module: INTC

A system sysgened without the SVC intercept feature tried to
process an SVC interception. R8 contains the address following
the call to the intercept routine.

Crash code: 300
Module: EXTM

A task event service routine is being dispatched, but the task
does not have a TQH. R9 contains the TCB address.

Crash code: 301
Module: EXTM

An attempt is made to dispatch a task event when there are none
available for dispatching. R9 contains the TCB address.

Crash code: 401*
Module: EXTM

The TCB address of the task being rolled in is not at the head of
the roll queue, or the TCB.STAT field of the task being rolled in
has a status bit turned on that is not valid when a task is being
rolled in. Register R9 contains the address of the TCB.

Crash code: 402*
Module: EXTM

RINQUE is trying to put a task on the rollin queue, but the task's TCB address is invalid. Register R9 contains the address of the TCB.

Crash code: 403*
Module: EXTM

RINDQUE is trying to remove a task from the rollin queue, but the task's TCB address is invalid. Register R9 contains the address of the TCB.

Crash code: 404
Module: EXTM

In a system without roll support, TMCKUTET found a task was being dispatched without any memory allocated to its impure segment. Register R9 contains the address of the TCB. Register RD contains the flag field TCB.STAT of the TCB.

Crash code: 405
Module: EXTM

In a system without roll support, TMCKUTET found the rollin pending bit on in the TCB. Register R9 contains the address of the TCB. Register RD contains the flag field TCB.STAT of the TCB.

Crash code: 500
Module: INITMSM

The operating system was overwritten by a misdirected DMA transfer into memory. Register D contains A(DCB), register 0 contains the last location overwritten, and register 2 contains the I/O bus address of the direct memory access (DMA) device.

Crash code: 501
Module: INITSUBS

An attempt was made to use two level translation, but the second level translation table was not specified. Register 4 contains A(CCB); register 6 contains the character being translated.

Crash codes 600 through 610 apply only to users of the Model 3200MPS System.

Crash code: 600
Module: APSV

ZAPWAIT found the task's status had APU waiting set, but the APU number was zero, or no APB could be found for the APU number, or the APB information did not indicate that it was waiting for the task.

Crash code: 601
Module: APSV

(SVC 13, APUINIT, APUESR.) The APB directory pointer located at X'CO' was not quadword aligned.

Crash code: 602
Module: APSV

(SVC 13, APUINIT, APUESR) The APB address obtained from the APB directory was not quadword aligned.

Crash code: 603
Module: APSV

GRABTASK could not find the APB for the APU number assigned to the task, or time-out occurred while attempting to access that APU's ready task queue.

Crash code: 604*
Module: APSV

RELAPU discovered an APU waiting for a task that does not have the APU waiting status set. In systems without safety-checking, the APU is simply restarted.

Crash code: 605
Module: APSV

APUINIT, APUESR, or RTSMDRVR discovered an invalid number of APUs.

Crash code: 608*
Module: APSV

TMCKAPU discovered that the task being dispatched was assigned a nonzero logical processing unit (LPU) number, but the SPT.LPMT pointer was zero.

Crash code: 609*
Module: APSV

TMCKAPU discovered that the task being dispatched was assigned an LPU number that was greater than LPMT.LPU.

Crash code: 60A*
Module: APSV

TMCKAPU discovered that the APU number found in the LPMT for the task's LPU assignment was greater than LPMT.APU.

Crash code: 60B*
Module: APSV

TMCKAPU discovered that the APB for the APU on which the task was to be dispatched could not be found, or if found, its APB.ID did not equal the expected APU number.

Crash code: 60C
Module: APSV

TMAPUFH discovered that the task's passback reason code (TCB.PRCA) indicated an SVC interrupt.

Crash code: 610
Module: APSV

RCVRESR was unable to access the CPU receive queue to process a task passback from an APU (queue lock timeout).

Crash code: 611*
Module: APSV

RCVRESR received a "passback to CPU" signal from an APU, but no TCB could be found on the CPU receive queue. Typically caused by a spurious interrupt from the RTSM, causing the last signal received to be re-read as a new signal.

Crash code: 801
Module: ITAM.M01

ITAM buffer management - End Buffer routine finds a CCB with zero
as the address of the current buffer. This is usually due to
improper management of the CCW buffer select bit in the CCB.

Crash code: 802
Module: ITAM.M01

During an attempt to form ring in ITAM timer chain, the CCB is
found to be already on the timer chain.

Crash code: 803
Module: ITAM.M01

DCB.ITB field is in illogical state. The field is being changed
incorrectly.

Crash code: 804
Module: ITAM.M01

Loss of buffer control using queued buffers. Internal queued
buffer count is greater than zero but no buffers exist.

Crash code: 805
Module: ITAM.M01

ITAM Buffer Management - RAW-Read-After-Write-Next-Buff routine
finds zero as the address of the current buffer.

Crash code: 806
Module: ITAM.M01

ITAM Buffer Management - Next-Buff routine entered but buffer
type is neither chained nor queued.

Crash code: 807
Module: ITAM.M01

ITAM Buffer Management - Next-Buff routine finds zero as the
address of the current buffer.

Crash code: 808
Module: ITAM.M01

ITAM Buffer Management - Next-Buff routine finds zero as the
address of the noncurrent buffer.

Crash code: 811
Module: ITFM.M00

CANITAM invoked on a non-ITAM system.

Crash code: 813
Module: CTM

Illegal response mode code.

Crash code: 814
Module: CTM

Illegal trap code from the CTM.

Crash code: 815
Module: CTM

Illegal FRMR reason code received.

Crash code: 816
Module: CTM

No allocated Drop Control Tables.

Crash code: 817
Module: CTM

DCB.NACD is less than the number of entries on the DCT chain.

Crash code: 819
Module: CTM

DCB.ONRT is not set up for retransmission.

Crash code: 820
Module: CTM

Illegal poll outstanding on line.

Crash code: 821
Module: CTM

Read pool not set up.

Crash code: 822
Module: CTM

Attempted write when write active.

Crash code: 823
Module: CTM

Read after write attempted with read active.   No read pool
present.

Crash code: 827
Module: CTM

Missing frame on the Internal Done Write (IDW) list.

Crash code: 828
Module: CTM

Frame out of sequence on internal done read (IDR) list.

Crash code: 830
Module: CTM

Problem returning buffer to user read buffer pool.   Either the
address of the buffer is invalid, or the address of the buffer is
equal to the address of the buffer at the bottom of the list.

Crash Code: 831
Module: CTM

Address translation problem in returning buffer to user.

Crash code: 832
Module: INITMSUP

An attempt was made to reenter the asynchronous terminal manager
kernel for a nonpolling extended function. Only polling extended
functions require and allow a second entry.

Crash code: 833
Module: INITMSM

An assignment to a drop on a multidrop line was   found
disconnected   while   attempting   to   process   an   I/O.   A
conversational I/O was attempted using a drop on a multidrop
line.

Crash code: 834
Module: HSUP

An attempt was made to process a logical unit that either is not
assigned to a drop or for which a drop does not exist.

Crash code: 835
Module: INITMSUP

An attempt was made to add to an internal SVC15 buffer that was
already full.

Crash code: F01
Module: EXIO

A leaf being disconnected from the current task is not queued to
the task TCB. Register R9 contains the address of the current
TCB. Register R15 contains the address of the leaf being
disconnected.

Bidirectional input/output control (BIOC) is a standard OS/32 terminal driver. Listed in this appendix are function control codes for the BIOC, the standard control characters generated by the use of the codes, and the functions performed. On terminals that do not generate standard control characters for any of the function keys, it is necessary to determine which key will produce the required control characters in order to invoke a desired function.

When a combination of control and ASCII keys cannot be accepted, BIOC rejects that combination and responds with a bell code. An example of this is a "cancel" request (CTRL-X) on a line that has no character on it. ASCII control characters for BIOC are not echoed (displayed to the console) to prevent confusion between BIOC functions and terminal functions.

ASCII Read Mode:

CTRL-A (SOH) Adjust Baud Rate

The baud rate adjust function must be enabled by the system programmer before CTRL-A can be used. When connection to a terminal is made over a dial-up line, the adjust baud rate mode is automatically entered.

To change the baud rate on a Perkin-Elmer Model 1200 terminal, for example, locate the front panel and remove the cover. It is important to know which baud rates have been made available to the terminal via strapping on the PASLA, 8-line COMM MUX, etc. When this is known, depress CTRL-A and then change the baud rate setting inside the panel, using the scale depicted on the inside of the panel cover (see Figure E-1). By depressing the carriage return (CR) key repeatedly, the user synchronizes communication at the new baud rate. BIOC then responds with an asterisk (*) and continues with the newly selected baud rate.

BAUD RATE	PROG. MODE	STOP BIT	DUPLEX	PARITY	AUTO TAB	INV. VID.
9600 7200 4800 2400 1800 1200 600 300 200 110 75	ON | OFF	ONE | TWO	FULL | HALF	SPACE MARK EVEN ODD	ON | OFF	ON | OFF

Figure E-1 Perkin-Elmer Model 1200 Mode Selectors

CTRL-B (STX) Backspace (Nondestructive)

This code causes the cursor to backspace one character each time the code is used. To be effective, CTRL-B cannot be entered at the first character position on a line. When the cursor has been backspaced to the desired character position, the line may be changed by typing the desired characters. All other characters backspaced over can be restored and the cursor brought back to the end of the line in one of two ways:

● CTRL-F, moves the cursor forward one character at a time

● CTRL-Z, "zooms" the cursor immediately to the end of the line

CTRL-C (ETX) Capture the Last Line Entered

Entering this code causes the last line entered (maximum of 80 characters) to be displayed on the console. By using CTRL-C repeatedly, character strings can be concatenated. If an insert or delete function is performed, the CTRL-C code is rejected and a bell sounds to remind you that the buffer has now been overwritten.   CTRL-C is also rejected if the display of data to the console has been suppressed by the use of CTRL-E.

CTRL-D (EOT) Device Control -- Echo Only

The next character entered after the CTRL-D code is echoed to the terminal but is not stored in the input buffer.   This function could be helpful, for example, if an auxiliary peripheral is used that requires certain control characters to be entered at the console. The CTRL-D code can prevent the peripheral control characters from being interpreted as program input.

CTRL-E (ENQ) Echo Toggle

Each entry of CTRL-E changes the current echo state from ON to OFF, or from OFF to ON. This means that data display to the console screen can be controlled. Suppression of data display is useful for entering passwords without others being able to observe them. All functions work with echo off except CTRL-C, CTRL-R, CTRL-W, CTRL-], CTRL-^, and CTRL-_. A CTRL-M (carriage return), buffer full, or CTRL-X turns echo back on. A CTRL-E is rejected if insert mode is in effect.

CTRL-F (ACK) Forward Space and Restore

This code is used to restore a line that has been backspaced over by the CTRL-B, CTRL-W, or CTRL-] code. After the cursor has been moved to the desired position and the correction has been made, CTRL-F moves the cursor forward one character position at a time until it reaches the end of the line. CTRL-F is rejected if there are no characters to be restored.

CTRL-H (BS) Backspace (Destructive)

This code is used to delete a character or characters. Unlike CTRL-B, however, any character(s) backspaced over by using the the CTRL-H code cannot be restored by using the CTRL-F or CTRL-Z codes and must be retyped. If they are not retyped, blank spaces appear in those character positions. CTRL-H is rejected if attempted at the first character position in a line. On most terminals the CTRL-H code can be generated by the "backspace" key.

CTRL-L (FF) Set Page Pause Line Count

To set the CRT screen display for a specific number of lines, the CTRL-L code is entered, followed by depressing the control key again with another ASCII character. The numeric value of the ASCII character sets the number cf lines to be displayed. To select a count for a 24-line CRT, enter the sequence: CTRL-L, CTRL-X (CTRL-X has a decimal value of 24).

The following table shows the proper combinations for line displays ranging from 1 to 24.

TABLE E-1 LINE DISPLAY COMBINATIONS

SEQUENCE	NUMBER OF LINES
CTRL-L CTRL-A	1
CTRL-L CTRL-B	2
CTRL-L CTRL-C	3
CTRL-L CTRL-D	4
CTRL-L CTRL-E	5
CTRL-L CTRL-F	6
CTRL-L CTRL-G	7
CTRL-L CTRL-H	8
CTRL-L CTRL-I	9
CTRL-L CTRL-J	10
CTRL-L CTRL-K	11
CTRL-L CTRL-L	12
CTRL-L CTRL-M	13
CTRL-L CTRL-N	14
CTRL-L CTRL-O	15
CTRL-L CTRL-P	16
CTRL-L CTRL-Q	17
CTRL-L CTRL-R	18
CTRL-L CTRL-S	19
CTRL-L CTRL-T	20
CTRL-L CTRL-U	21
CTRL-L CTRL-V	22
CTRL-L CTRL-W	23
CTRL-L CTRL-X	24

Each display of the requested number of lines is terminated with a bell sound. At this point the user may continue to the next page by entering a carriage return (CR). This causes the same number of lines to appear; each CR, in fact, produces that number of lines until the page pause line count is changed. To change the count, terminate write by entering ESC or Break, and enter a different sequence for the desired new line count (e.g., CTRL-L CTRL-O = 15 lines, etc.).

To cancel the page pause mode, use the sequence CTRL-L, CTRL-@, or CTRL-N. If the page pause mode is not terminated within 5 minutes, BIOC automatically continues output to prevent the terminal from being permanently tied up.

CTRL-M (CR) Terminate Read

This function is a carriage return. Entering CTRL-M indicates to BIOC that read should be terminated. If CTRL-M is entered at a location other than the end of the line, BIOC performs a zoom to the end of the line (EOL) before storing the carriage return and terminating the read request.

CTRL-N (SO) Neutralize Selected Options Back to Default

This code is entered to reset options to their default values. CTRL-N can be entered during read operations, during write operations, or between read and write operations. Entering CTRL-N performs the following functions:

- resets page pause to zero

- resets backspace prompt character to CTRL-H

- resets ASCII read prompt character to sysgen default

- resets backspace and CR/LF protocol to sysgen default

- resets output mode to print-on state

CTRL-O (SI) Toggle Output Between Print-on and Print-off

To suppress output in the write mode, CTRL-O is used. To resume output, this code is used again. Alternately depressing CTRL-O causes output to terminate and resume; hence, the "toggle" characteristic. When using CTRL-O to select the print-off mode, a prompt can be immediately received by a terminate read (CTRL-M). If this is not done within 15 seconds after output ceases, BIOC prompts and reinstates the print-on mode automatically. The print-on mode is also reinstated upon successful completion of a read request, or upon entering CTRL-N for a neutralize function.

CTRL-P (DLE) Set ASCII Read Prompt Character

By entering CTRL-P and any ASCII character, that character becomes the designated prompt. When making the selection, the ASCII character is not displayed to the console, but is output by BIOC upon receipt of an ASCII read request. The read prompt function can be turned off by the sequence CTRL-P CTRL-X. To reset the ASCII read prompt character to the sysgen default, enter CTRL-N.

CTRL-Q (DC1) Removed from Input to Allow X-ON/X-OFF Flow Control

CTRL-R (DC 2) Reprint Entered Line

When this code is entered, the current cursor location within the line determines the number of characters that are reprinted on the next line. All characters to the left of the cursor including blank spaces, are reprinted. The CTRL-R function is rejected if the echo state is not in effect (see CTRL-E).

The CTRL-R function is especially useful for hardcopy terminals where corrections are made over the existing typed lines. To view a "clean" line after all corrections have been made, CTRL-R is used.

CTRL-S (DC 3) Removed from Input to Allow X-ON/X-OFF Flow Control

CTRL-T (DC4) Single Character Transparent Mode

The use of this code allows the entry of function control characters into the input buffer. The next character entered after a CTRL-T is entered directly into the input buffer.

CTRL-W (ETB) Word Backspace (Nondestructive)

CTRL-W causes the cursor to be backspaced (nondestructively) to the nearest nonalphabetic character. Thus, CTRL-W allows the cursor to backspace over one complete word, rather than one character, as with CTRL-B. Words backspaced over may be restored by the use of CTRL-F or CTRL-Z. CTRL-W is rejected if attempted at the beginning of a line.

CTRL-X (CAN) Cancel Current Input Line

All characters previously entered on the current line are deleted upon use of the code. Characters may not be restored with the CTRL-F or CTRL-Z functions. If no characters are on the line, CTRL-X is rejected. CTRL-X turns echo back on if it has been turned off with CTRL-E.

CTRL-Z (SUB) "Zoom" to Furthest End of Line

CTRL-Z can be used to restore a line that has been backspaced over by CTRL-B, CTRL-W, or CTRL-]. CTRL-Z causes the cursor to "zoom" to the end of the line, but is rejected if there are no characters to be restored.

CTRL-] (GS) Backward Character Search (Nondestructive)

This code serves to locate a specific character on the current line. For example, to find the character $, enter CTRL-]$. BIOC backspaces until the first $ is found. To find any additional dollar signs on the same line, the code must be entered again for each time the $ symbol appears. Characters backspaced over may be restored by using CTRL-F or CTRL-Z. CTRL-] will be rejected if attempted at the beginning of the line.

CTRL-∧ (RS) Toggle Between Insert-on and Insert-off

Each CTRL-∧ toggles from insert-on to insert-off or from insert-off to insert-on. When the insert mode is selected, characters typed are inserted to the left of the character currently at the cursor position. The insert mode may be selected only when the echo state is in effect and the cursor is positioned at a location other than end of the line. Insert mode is terminated by another CTRL-∧ or by any command that moves the cursor to the end of line (e.g., CTRL-Z). The CTRL-C and CTRL-E functions are not valid while in the insert mode. All other functions are valid if the cursor is not in motion. All data entered while the cursor is in motion is ignored until the cursor has stopped.

CTRL-_ (US) Delete Character

Each CTRL-_ deletes the character currently at the cursor position. The delete code is valid only when the echo state is in effect and the cursor is positioned at a location other than end of line. Characters entered while the cursor is in motion are ignored.

Write Mode:

If the following codes are entered while BIOC is writing to the console, the described actions occur:

BREAK

   This key terminates write with a Break status.

ESC

   This key terminates write with a Break status.

CTRL-Q

This code resumes write after write has been suspended by
CTRL-T or CTRL-S functions.

CTRL-R

This code resumes write after write has been suspended by
CTRL-T or CTRL-S functions.

CTRL-S

This code suspends write until write is resumed by CTRL-R or
CTRL-Q, or until the BREAK or ESC key is depressed.

CTRL-T

This code suspends write until write is resumed by CTRL-R or
CTRL-Q, or until the BREAK or ESC key is depressed.

INDEX

PUBLICATION COMMENT FORM

Please use this postage-paid form to make any comments. suggestions. criticisms, etc. concerning this publication.

From _____   Date _____

Title _____   Publication Title _____

Company_____   Publication Number _____

Address _____

_____

_____

FOLD   FOLD

Check the appropriate item.

☐ Error   Page No. _____   Drawing No. _____

☐ Addition   Page No. _____   Drawing No._____

☐ Other   Page No._____   Drawing No. _____

Explanation:

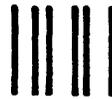CUT ALONG LINE

Fold and Staple
No postage necessary if mailed in U.S.A.

6434

BUSINESS REPLY MAIL

FIRST CLASS   PERMIT NO. 22   OCEANPORT, N.J.

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

POSTAGE WILL BE PAID BY ADDRESSEE

PERKIN-ELMER

Computer Systems Division
2 Crescent Place
Oceanport, NJ 07757

TECH PUBLICATIONS DEPT. MS 322A

D O C U M E N T A T I O N   C H A N G E   N O T I C E

The purpose of this documentation change notice (DCN) is to provide a quick and efficient way of making technical changes to manuals before they are formally updated or revised.

The manual affected by these changes is:

48-030 F00 R01   OS/32 Operator Reference Manual

● **Page 3-148**

In the Parameters column, add IMAGE directly under VFC.   Then change the last sentence of the VFC paragraph to:

If IMAGE is specified, there is no VFC for the device assigned to the specified lu.

● **Page 3-148**

In the Parameters column, add NOIMAGE   directly   above   NOVFC. Then change the paragraph to:

turns the VFC control option   or   IMAGE   option   off   for   the assigned lu.   NOVFC is the default option.

● **Page 3-148**

Add the following parameters and descriptions before COPIES=:

CHECKPOINT   turns on checkpointing for   the   assigned   lu. This   is   the   default   option.   The   global checkpoint option must be on.

NOCHECKPOINT   turns off checkpointing for the assigned lu.