MP/M II T.M. Operating System

Release 2.1

Release Notes

Digital Research
P.O. Box 579
160 Central Avenue
Pacific Grove, CA 93950
(408) 649-3896
TWX 910 360 5001

Dear MP/M II User:

Digital Research has developed the MP/M II T.M. operating system in response to numerous customer requests to add file sharing capability to MP/M Release 1.1. The design of MP/M II is a reflection of our goal to provide you with a state of the art operating system that can be configured for a wide variety of computer hardware.

This shipment contains the version 2.1 release of our MP/M II operating system. We have been pleased with the response to MP/M II Release 2.0 and hope to see comparable response to MP/M II Release 2.1 regarding design, possible extensions, and errors in implementation. We hope to maintain the same level of confidence that the computer industry has had in our CP/M ® operating system.

On the basis of our experience and the experience of MP/M II users, we estimate it requires less than a week to implement a simple polled MP/M II on a computer that has a running version of CP/M Release 2.2. Implementing a highly optimized MP/M II system with full interrupts and bank switched memory can require several weeks. Of course, the time to perform such a reconfiguration will vary widely depending on the experience of the programmer and the complexity of the hardware.

Note: Make sure you use the SET or STAT command to make the USER.PRL file into a system file.

Contact the Digital Research Technical Support staff (408) 375-6262 if you experience difficulties reconfiguring MP/M II. By sending in your registration card you can insure that we will mail MP/M II application notes and patches that correct implementation errors.

Sincerely,

TECHNICAL SUPPORT

MP/M II T.M. Operating System
Release 2.1
Extended File Locking

Addendum to the MP/M II Operating System Programmer's Guide

Compiled January 1982

Extended file locking is a new facility implemented in release 2.1 of MP/M II T.M. . Extended file locking enables a process to maintain a lock on a file even after the file is closed. This facility allows a process to rename, set the attributes, or delete a file without having to contend with the possibility of interference from other processes after the file is closed. Also, a process can reopen a file with an extended lock and continue normal file processing. For example, a process can open a file, perform file operations on the file, close the file, rename the file, reopen the file under its new name, and proceed with file operations, without ever losing the file's lock list item and control over the file.

Extended file locking is only available to files that are opened in the default open mode (locked mode). To extend a file's lock, set interface attribute F6' when closing the file. This attribute is only interrogated by the Close function when it is closing a file permanently. Thus, interface attribute F5' must be reset when the close call is made. Also, if a file has been opened N times (more than once), this attribute is only interrogated when the file is closed for the Nth time.

To maintain an extended file lock through a Rename File call or a Set File Attributes call, set interface attribute F5' of the referenced FCB when making the call. This attribute is only honored for extended file locks, not normal locks. Setting attribute F5' also maintains an extended file lock for the Delete File function, but setting this attribute also changes the nature of the Delete function to an XFCB-Only delete. If successful, all three of these functions delete a file's extended lock item when with attribute F5' reset. On the other hand, if they return with an error code, the extended lock item is not deleted.

A standard open call can be made to resume file operations on a file with an extended lock. The open mode, however, is restricted to the default locked mode. The following list illustrates uses of extended locks.

- Open file EXLOCK.TST in locked mode.

- Perform file operations on the file EXLOCK.TST using the open FCB.

- Close file EXLOCK.TST with interface attribute F6' set to retain the file's lock item.

- Use the Rename File function to change the name of the file to EXLOCK.NEW with interface attribute F5' set to retain the file's extended lock item.

- Open the file EXLOCK.NEW in locked mode.

- Perform file operations on the file EXLOCK.NEW using the opened FCB.

- Close file EXLOCK.NEW with interface attribute F6' set to retain the file's lock item.

- Set the Read-Only attribute and release the file's lock item by using the Set File Attributes function with interface attribute F5' reset. At this point, the file EXLOCK.NEW becomes available for access by another process.

MP/M II$^{T.M.}$ Operating System
Release 2.1
Compatibility Attributes

Addendum to the <u>MP/M II Operating System Programmer's Guide</u>

Compiled February 1982

The MP/M II$^{T.M.}$ file system introduced some new restrictions relating to file operations that were not present in MP/M$^{T.M.}$ 1.1 or CP/M®. For example, if a process opens a file in the default mode (locked), MP/M II does not allow other processes on the system to open, delete, or rename the file until the process opening the file either closes the file or terminates. In addition, MP/M II does not allow a process to perform file operations with an FCB that has not been activated by a successful open or make operation, or with an FCB that has been deactivated by a close operation. These restrictions protect an MP/M II user from interference from other users on his open files. To illustrate, it is this protection that enables an MP/M II user to edit a file with the assurance that another user cannot delete or modify his file during his edit session.

The new restrictions added to MP/M II provide file security when multiple users are running the system. The preceding example describes restrictions required to prevent collisions on file activity between independent processes. Another new MP/M II restriction sets limits on how a process can modify open FCBs. These limits are enforced by checksum verification of open FCBs and protect the integrity of the MP/M II file system from corrupted FCBs. Note that the new MP/M II restrictions are not intended to protect a user from his own actions. Instead, they ensure that the activity of one user does not adversely affect other users on the system.

Generally, the new MP/M II file system restrictions create little difficulty for new application development. In fact, they enforce good programming practice. However, because of these new restrictions, some CP/M and MP/M software written before MP/M II's release does not run on MP/M II. In addition, multiple copies of some software do not run because the default open mode for MP/M II is a locked mode in which only one process can open a file.

To address these problems, Digital Research has added compatibility attributes to MP/M II, Release 2.1. The compatibility attributes are defined as attributes F1' through F4' of program files. A new GENSYS option determines whether the attributes are to be activated. If activated, the Command Line Interpreter (CLI) interrogates these attributes during program loading and modifies the MP/M II ground rules for the loaded program as described below.

Note that the compatibility attributes should not be used with new software. They are intended for use with working software developed for CP/M and MP/M 1.1. This especially applies to compatibility attribute F4', which disables FCB checksum verification on read and write operations. Use this attribute sparingly and only with programs that are known to work.

COMPATIBILITY ATTRIBUTE DEFINITIONS

F1' MP/M 1.1 Default Open. Processes running with this attribute have all files opened in locked mode marked as Read-Only in the System Lock List. This allows all processes with this attribute set to read and write to common files with no restrictions. There is, however, no record locking provided. In addition, this attribute also allows a process to write to a file opened by another process in Read-Only mode. To be safe, all static files such as program and help files should be made Read-Only when this compatibility attribute is used.

F2' Partial Close default. Processes running with this attribute have their default close mode changed from permanent close to partial close. This attribute is for programs that close a file to update the directory but continue to use the file. Note that MP/M II assumes a process has finished with a file when the number of closes issued to the file equals the number of opens. A side effect of this attribute is that files opened by a process are not released until the process terminates. It might be necessary to set the System Lock List parameters to high values when using this attribute.

F3' Ignore Close Checksum Errors. This attribute changes the way Close Checksum errors are handled for a process. Usually, a message is printed on the console, and the process is terminated. When this attribute is set and a checksum error is detected during a close operation, the file is closed if a lock list item exists for the file. Otherwise, an unsuccessful close error code is returned to the calling process.

F4' Disable FCB Checksum verification for read and write operations. Setting this attribute also sets attributes F2' and F3'. This attribute should be used carefully because it effectively disables MP/M II's file security. Use this attribute only with software that is known to work.

PROCEDURE FOR USING THE COMPATIBILITY ATTRIBUTES

1) Answer yes to the GENSYS question "Enable Compatibility Attributes (N) ?".

2) Use the MP/M II Utility SET to set the desired combination of compatability attributes in the program name.

EXAMPLES:

```
0A>SET filespec [F1=on]
0A>SET filespec [F1=on,F3=on]
0A>SET filespec [F4=on]
```

If you have a program that runs under CP/M or MP/M 1.1 but does not run properly under MP/M II, use the following guidelines to select the compatibility attributes to set for the program.

1) If the program ends with the message, "File Currently Opened" when multiple copies of the program are run, set compatibility attribute F1'. As an alternative, you might consider placing all common static files under User 0 with the SYS and R/O attributes set.

2) If the program terminates with the message, "Close Checksum Error", set compatibility attribute F3'.

3) If the program terminates with an I/O error, try running the program with attribute F2' set. If the problem still occurs, try attributes F2' and F3'. If the problem still persists, then try attribute F4'. Use attribute F4' only as a last resort.

It might be necessary to increase the GENSYS parameters that set the maximum number of files a process can open and the size of the System Lock List when using compatibility attributes F2' and F4'. This might be required because both default to partial closes. As a result, system lock list entries consumed by opening files are not released until the process ends. Generally, if a process ends with the message "No Room in System Lock List" or "Open File Limit Exceeded", it usually indicates that the above GENSYS parameters need to be increased. Another option is to patch in a BDOS Free Drive call at a point in the program where no files are active. Note that a Free Drive call specifying all drives, purges all file system lock entries belonging to the calling process.

When GENSYS activates compatibility attributes, the Command Line Interpreter copies the settings for attributes Fl' through F4' of the filename of the loaded program into byte lDH of the process descriptor as shown below:

PROCESS DESCRIPTOR BYTE lDH

(Bits defined 7-0 high-order to low-order)

Bit 7 set = Fl
Bit 6 set = F2
Bit 5 set = F3
Bit 4 set = F4

MP/M II T.M. Operating System
Release 2.1
Programming Guidelines

Addendum to the MP/M II Operating System Programmer's Guide

Compiled January 1982

This guideline provides additional discussion on the information presented in the MP/M II T.M. Operating System Programmer's Guide. In particular, this document emphasizes those areas of MP/M II where restrictions exist that did not exist in versions 1 and 2 of CP/M® and version 1 of MP/M T.M.. The intent is to enable the MP/M II application programmer to avoid potential problems with new software. As a prerequisite, the reader should be familiar with the material presented in the MP/M II Operating System Programmer's Guide.

1) Always use the following sequence when performing file operations that require an open file. Under MP/M II, these operations are the BDOS read, write, lock, and unlock record commands.

- Activate a file's FCB with a BDOS Open or Make function call before using the FCB in a file operation. Verify that the Open or Make operation was successful. MP/M II only accepts FCBs activated by a successful Open or Make call for open file operations. If an FCB that has not been activated is used, MP/M II returns a checksum error.

- Perform all file operations using activated FCBs. Note that MP/M II does not deactivate an activated FCB when it returns error codes for file operations. Generally, only the current record and random record fields of an activated FCB should be modified. In addition, all file operations with an activated FCB must be made under the user number that was in effect when the FCB was activated. A similar restriction applies to activated FCBs that specify the default drive. All file operations specifying such an FCB must be made under the current drive that was in effect when the FCB was activated. Item 3 in this list covers the complete rules regarding activated FCB modification.

- If a process has completed file operations on a file but still has a significant amount of processing left to do, the file should be closed. This applies even if the file was not modified. With some exceptions, the lock list entry associated with a file in the system lock list is not released until a file is permanently closed (MP/M II Operating System Programmer's Guide see Section 2.2.9.) MP/M II restricts access to a file by other processes while a lock list item for the file resides in the system lock list. It is not necessary to close input files if a process is about to end. At termination, all lock items belonging to a process are released. Output files, however, must always be closed or data might be lost. Note that a successful permanent close operation deactivates the FCB and removes the file's item from the system lock list. If the deactivated FCB is used in a subsequent open file operation, MP/M II returns a checksum error.

2) If a process opens the same file more than once, a matching number of close commands must be issued to the file to remove the file's lock list item from the system lock list. Thus, if a file has been opened N times, the first N-1 close operations issued to the file default to partial close operations. Only the last close, close operation N, is interpreted as a permanent close. By definition, a permanent close is a close operation that removes the referenced file's item from the system lock list. Note that only one lock list item is allocated in the system lock list for a file regardless of the number of FCBs a process has opened for the file.

3) The following list specifies how an activated FCB can be changed without affecting the FCB checksum. MP/M II returns a checksum error code and does not perform the requested operation if an FCB with an invalid checksum is used in an open file operation.

- FCB(0) cannot specify a new drive.
- With the exception of interface attributes F5' and F6' for the BDOS Close function, FCB(1) through FCB(11) cannot be changed.
- The high-order 3 bits of FCB(12) cannot be changed. The low-order 5 bits can be changed. Note that when a file is opened in the default open mode (locked mode), the high-order 3 bits of this FCB field are set to zeros.
- FCB(13) cannot be changed.
- FCB(14) and FCB(15) can be changed.
- FCB(16) through FCB(31) cannot be changed.
- FCB(32) through FCB(35) can be changed.

If compatibilty with future releases of MP/M and CP/M is a requirement, programs should restrict open FCB modification to the FCB fields 32 through 35. In particular, Digital Research does not support techniques that involve modifying fields 12, 14, and 15 of open FCBs.

4) Processes that access a printer must issue a Detach List device to free the printer before another process can use the printer. If the Detach List call is not made, a process that accesses a printer continues to own the printer until it ends.

5) CP/M programs that create submit files for chaining must be modified to work under MP/M II. MP/M II requires a different filename for submit files, that includes the originating console number, and requires that a submit flag be set in the System Data Page. The technique for creating and executing submit files is described in MP/M II Application Note 07. Note that MP/M II also has a Program Chain (Function 47) command that provides an efficient mechanism for program chaining.

6) CP/M programs that make direct BIOS calls for disk I/O do not work under MP/M II. MP/M II does support direct XIOS calls for the console and printer, but not to the disk. If programs must make direct XIOS disk calls, a technique **strongly** discouraged in a multiuser environment, two levels of indirection must be used to obtain the real XIOS jump table address. The second level of indirection is required because an intercept table handles the console and printer.

   The following two steps should be performed in a program before making direct XIOS calls to a disk. The first step is to make a BDOS Write Protect Disk (Function 28) call to the disk to ensure that no other process has open files on the disk. Secondly, the MXDisk mutual exclusion queue message should be read to prevent other programs from making BDOS disk function calls while your program is making direct XIOS calls. After completing your direct XIOS calls, write back the MXDisk message and then reset the drives you have set to Read-Only.

7) The following procedure is a protocol that multiple processes can use to coordinate record update and addition operations to a shared file. Each process must open the shared file in unlocked mode. This procedure also assumes that records containing binary zeros are null records.

- Attempt to lock the record.
- If the lock attempt fails because another process has locked the record, delay and repeat the procedure.
- If the lock attempt fails because the record does not exist in the file, add a record initialized to binary zeros to the file with the BDOS Write Random with Zero Fill command and repeat the procedure. Note that files opened in unlocked mode are extended in block units and not in record units as is the case for files opened in the default locked mode.
- If the lock attempt succeeds, read the record, update it, and then unlock it.

8) Multiple FCB I/O is a technique that involves opening each extent for a file independently and maintaining them in a table in memory. Then random I/O is handled by selecting the proper FCB from the table, setting the current record field to the proper record number within the extent, and making a sequential Read or Write command. When processing is completed, each FCB is closed. The maximum file size that can be accessed with this technique is 512K bytes. This limits the maximum table size to 32 FCBs. Note that this technique provides a method of performing random I/O that is compatible with CP/M 1.4.

Multiple FCB I/O has to be performed carefully under MP/M II because of the restrictions MP/M II places on file operations to provide file security. Generally, an FCB should not be used in file I/O unless it has been activated and it should not be modified while it is activated (see items 1 and 3). In addition, the number of opens and closes issued to a file is important (see item 2). Note that all 32 bytes of each extent's FCB should be maintained in the open FCB table. Also, verify that interface attribute F8' is set to 1 in all FCBs if the first FCB has F8' set to 1. F8' set to 1 indicates the file was opened under user 0 although the current user number is nonzero (see Function 15 in the MP/M II Operating System Programmer's Guide).

MP/M II$^{T.M.}$ Operating System Release 2.0
Application Note 01, 9/14/81

Compiled September 1981

SUPPRESSING THE MP/M$^{T.M.}$ LOADER DISPLAY

Applicable products and version numbers: MP/M II$^{T.M.}$ Release 2.0

Program: MPMLDR.COM

When the MP/M II loader reads the MPM.SYS file, it displays a load map on console #0. In some applications you might want to suppress this display.

To suppress the load map display on console #0, type the following RET instruction into the LDRBIOS.ASM file using any standard editor. The RET instruction replaces the console output code.

```
; Loader BIOS jump vector:

        ...
        jmp     conout
        ...

conout:
        ret
```

Assemble LDRBIOS.ASM to create LDRBIOS.HEX. Integrate the new LDRBIOS.HEX file into the MPMLDR.COM file according to instructions provided in the MP/M II Operating System System Guide. Then, update the system tracks of the boot disk with the new loader.

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{T.M.}$ Operating System Release 2.0
Application Note 02, 9/14/81

SETTING AND RESETTING THE RAW CONSOLE I/O MODE

Applicable products and version numbers: MP/M II$^{T.M.}$ Release 2.0

Some application programs require raw input from the console. Raw input implies that the operating system takes no action on special characters, such as CTRL-C.

Execute the following code to place an application program into a raw console input mode.

```
MVI  C,9CH
CALL XDOS    ; get process descriptor address
LXI  D,6
DAD  D
MOV  A,M
ORI  80H     ; turn 'on' the high-order bit of first
MOV  M,A     ;  character in the process name
...
```

Execute the following code to exit the raw console input mode.

```
...

MVI  C,9CH
CALL XDOS    ; get process descriptor address
LXI  D,6
DAD  D
MOV  A,M
ANI  7FH     ; turn 'off' the high-order bit of first
MOV  M,A     ; character in the process name
...
```

Functions 3, 4, and 6 place the system into raw console input mode. All other console I/O functions reset the system to normal console input mode.

Raw console input mode can cause problems. You cannot abort a process running in raw mode because the system ignores all control characters. To abort a process, use Function 11 before using any disk I/O functions. Function 11 returns the system to normal console mode.

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{T.M.}$ Operating System Release 2.0
Application Note 03, 9/14/81

CHANGING PRL FILE MINIMUM BUFFER SIZE REQUIREMENTS

Applicable products and version numbers: MP/M II$^{T.M.}$ Release 2.0

You might want to allocate a larger default buffer for a program such as the editor. You can change the minimum buffer size requirements for PRL files. The following procedure demonstrates how to change the minimum buffer size requirements for ED from 4k to 8k bytes.

```
0A>ddt ed.prl
[MP/M] DDT VERS 1.1
NEXT  PC
2300 0100
-s104
0104 00 00
0105 10 20
0106 .
-v2300
0044
-ied.prl
-w44
-g0
```

Bytes 4 and 5 of the PRL header record (relative to the base) contain the low- and high-order bytes for the minimum buffer size specification.

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{\text{T.M.}}$ Operating System Release 2.0
Application Note 04, 9/14/81

ACCESSING THE INTERNAL MP/M II$^{\text{T.M.}}$ TOD

Applicable products and version numbers: MP/M II Release 2.0

Some application programs might require access to the internal MP/M II time and date fields to set initial values. Execute the following code sequence at the end of your MP/M$^{\text{T.M.}}$ XIOS system initialization procedure. Place the code at the end because the XDOS call to obtain the system data page address might cause interruptions.

```
...

MVI  C,9AH
CALL XDOS       ;obtain the system data page address
                ;*** warning ***
                ;the XDOS call enables interrupts

LXI  D,00FCH
DAD  D          ; hl -> pointer -> TOD
MOV  E,M
INX  H
MOV  D,M        ; de -> TOD

...
```

The assembly language subroutine TODCNV.ASM distributed on the MP/M II release disk converts from ASCII string representation of the time and date to MP/M II internal time and date format.

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II T.M. Operating System Release 2.0
Application Note 05, 9/14/81

DMA DISK CONTROLLERS WITH BANKED MEMORY SYSTEMS

Applicable products and Version Numbers: MP/M II T.M. Release 2.0

Be extra careful with bank switched memory systems that have Direct Memory Access disk controllers. Bank switching is not allowed during a transfer of data from the disk controller to a target bank.

DMA from the disk controller is obtained through common memory, then copied from common memory into the user buffer that you wanted. Sectors larger than 128 bytes are placed in a common memory buffer. The specified sector is then transferred to the target buffer. This is a reasonable technique in systems where deblocking is required.

Use the following procedure if DMA is to occur directly into the user buffer bypassing common memory. Set a DMA active flag to true before each DMA operation. Reset the flag following each operation.

```
...

MVI  A,FFH
STA  DMACTVE

;  initiate DMA operation

;  perform flag wait or poll for operation complete

XRA  A
STA  DMACTVE
...
```

Place the following code sequence in the XIOS select memory procedure to ensure that the bank cannot be switched during a DMA operation:

```
SELMEMORY:
 ...

LDA  DMACTVE
ORA  A
JZ   OKTOSWITCH      ; jump if not in DMA operation
 ; Next, the bank to be switched can be
 ; compared with the current bank.  If
 ; it matches, the DMA operation will not be affected.
JZ   OKTOSWITCH      ; no bank change required
```

```
        ; A  new  bank is  specified and a DMA operation is  in
        ;  progress.  A busy wait must now be performed to wait
        ;  until the DMA operation is complete.

        ; *** warning ***

        ; The selmemory call is made from inside the dispatcher
        ;   therefore interrupts are disabled and nothing  must
        ;   be done that could force a dispatch.

       BUSYWAIT:
       IN   DMASTATUSPORT  ; This is a "BUSY-WAIT" !
       ANI  DMADONE
       JZ   BUSYWAIT       ; loop until the DMA is complete
```

Place the following code into the remaining select memory procedure.

```
       OKTOSWITCH:
          ...

          ...

       RET
```

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{T.M.}$ Operating System Release 2.0
Application Note 06, 9/14/81

USING THE SEND CLI COMMAND XDOS FUNCTION

Applicable products and version numbers: MP/M II$^{T.M.}$ Release 2.0

Use of the Send CLI Command XDOS Function can effectively implement a menu driven application program. The following steps outline use of the SEND CLI XDOS Function.

1) Change the priority of the calling process so that it is higher (actually a lower value) than the TMP.

2) Obtain the console number of the calling process.

3) Assign the console to the Command Line Interpreter.

4) Issue the send CLI command function call.

5) Issue an ATTACH console function to get the console back after the initiated process has terminated.

6) Restore the priority of the calling process to its original value (usually 200).

Segments of a menu driven program named MENU appear in the following example.

```
;
; XDOS Function Equate Table
;
setpriority     equ     145
attachconsole   equ     146
assignconsole   equ     149
sendCLIcommand  equ     150
getconsole      equ     153

MENU:
        ...

        mvi     e,190
        mvi     c,setpriority
        call    BDOS            ;set priority to 190
        mvi     c,getconsole
        call    BDOS            ;get console # in A reg
        sta     AssignPB        ;fill in
        sta     CLIcommand+1    ;       console fields
        lxi     d,AssignPB
        mvi     c,assignconsole
```

```
             call    BDOS            ;assign console to CLI
             inr     a
             jz      cannotassign    ;assign failed
             lxi     d,CLIcommand
             mvi     c,sendCLIcommand
             call    BDOS            ;send CLI command
             mvi     c,attachconsole
             call    BDOS            ;attach console
             mvi     e,200
             mvi     c,setpriority
             call    BDOS            ;set priority back to 200
             ...

    AssignPB:
             db      $-$             ;console number
             db      'cli     '      ;name (cli is lower case)
             db      0
             ...

    CLIcommand:
             db      0               ;default disk / user code
             db      $-$             ;console number
             db       this is an ASCII string terminated with a
                      null that is exactly as you would run the
                      program from the console. e.g.
                      'PIP LST:=MYPROG.LST[PT8]',0
             ...
```

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{T.M.}$ Operating System Release 2.0
Application Note 07, 9/14/81

CREATING A SUBMIT FILE FROM AN APPLICATIONS PROGRAM

Applicable products and version numbers: MP/M II$^{T.M.}$ Release 2.0

The following procedure shows you how to create a submit file from an applications program and force its execution. The procedure to terminate a submit file job is included.

1) Obtain the temporary file drive from the system data page.

2) Obtain the console number at which the program is executing.

3) Create the n.SUB file. Use n to specify the console number.

4) Set the appropriate submit flag in the array to on. The array is contained in the system data page.

```
;
; BDOS / XDOS Function Equate Table
;
closefile       equ     16
searchfirst     equ     17
deletefile      equ     19
makefile        equ     22
getconsole      equ     153
getsysdatadr    equ     154
subflgofst      equ     128
    ...
    ...
   mvi     c,getsysdatadr
   call    BDOS
   lxi     d,196            ; temp file drive offset
   dad     d
   mov     a,m
   sta     FCB
   mvi     c,getconsole
   call    BDOS
   sta     console
   adi     '0'
   sta     FCB+2            ; put console # in fname
   lxi     d,FCB
   mvi     c,searchfirst
   call    BDOS             ; see if file there
```

```
        inr     a
        jz      nofile
        lxi     d,FCB
        mvi     c,deletefile
        call    BDOS            ; delete old version first
                nofile:
        lxi     d,FCB
        mvi     c,makefile
        call    BDOS            ; make the $n$.SUB file
         ...
;
; Now, write the records into the the $n$.SUB file as
;  follows:
;              -one line of the submit file per record
;
;              -last record first (i.e. in reverse order
;                that they are to be executed
;
;              -each record in the following form:
;
;              [CNT][ASCII command line][NULL]
;                 where:  CNT = # chrs in cmnd ln, 1 byte
;                         ASCII command line <= 125 chrs
;                         NULL = zero, 1 byte
;
         ...
         ...
        lxi     d,FCB
        mvi     c,closefile
        call    BDOS            ; close the $n$.SUB; file
        mvi     c,getsysdatadr
        call    BDOS            ; get system data page adr
        lda     console         ; retrieve the saved con #
        adi     subflgofst      ; add offset to base of
        mov     e,a             ;  submit flag array
        mvi     d,0
        dad     d               ; DE = .submitflag(console);
        mvi     m,0ffh          ; set 'on' submit flag
         ...
; terminate the program
        ...
FCB:
        DB      1               ; disk drive, usually A:
        DB      '$n$     '      ; filename
        DB      'SUB'           ; filetype
        DB      0               ; file extent
        DS      20              ;  remainder of FCB
console:
        ds      1               ; temp loc for console #
         ...
```

Terminate the operation of a submit job by zeroing a submit flag located in the SYSTEM DATA PAGE region of memory. To locate and zero the submit flag for a console use the following code procedure.

```
;
;  XDOS Function Equate Table
;
getconsole      equ     153
getsysdatadr    equ     154
subflgofst      equ     128
         ...
         ...
        mvi     c,getconsole
        call    BDOS                ; get console #
        push    psw                 ; save console #
        mvi     c,getsysdatadr
        call    BDOS                ; get system data page adr
        pop     psw                 ; restore console #
        adi     subflgofst
        mov     l,a                 ; hl = address of sub. flag
        mvi     m,0                 ; zero submit flag
         ...
```

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II T.M. Operating System Release 2.0
Application Note 08, 9/14/81

FILE SHARING

Applicable products and version numbers: MP/M II T.M. Release 2.0

Multiple users can share files using the MP/M II file system. An applications program such as Wordstar® requires that files be open while the program is running. Multiple users of the application will need to share the open files. Usually under MP/M II, sharing of files causes problems if the applications program is not structured to open files in Read-Only mode. The default mode for the open function is locked mode which prevents the sharing of files. Files are opened in locked mode for earlier versions of both CP/M® and MP/M T.M. as well.

To enable file sharing, place all files to be shared under USER 0 on the default disk. Using the SET utility, assign the attributes SYS (System) and RO (Read-Only) to the files. The BDOS opens the file in Read-Only mode regardless of which mode the open function specified. An example is shown below.

```
OA>set wsmsgs.com [SYS,RO]
```

Licensed users are granted the right to inlude these enhancements in MP/M II Release 2.0 software.

MP/M II T.M. Operating System Release 2.0
Application Note 09, 9/14/81

PROGRAM CONTROL OF THE CONTROL-P SWITCH

Applicable products and version numbers: MP/M II T.M. Release 2.0

An applications program might need to echo console I/O to the printer while under program control. Use the following procedures to set and clear the CTRL-P flags. The array of flags is located at the SYSTEM DATA PAGE address + 126.

Setting CTRL-P Flag

```
          mvi     c,9ah       ; Get System Data Page address
          call    BDOS
          lxi     d,126
          dad     d           ; add 126 to Sys. Data Page addr.
          mov     e,m
          inx     h
          mov     d,m         ; DE = addr. of CTRL-P array
          push    d
          mvi     c,0a4h      ; Get List Number
          call    BDOS
          mov     e,a
          mvi     d,0
          pop     h
          dad     d
          mvi     m,0ffh      ; set CTRL-P flag
          ...                 ; cons. I/O is echoed from now on
```

Clearing CTRL-P Flag

```
          mvi     c,9ah       ; Get System Data Page address
          call    BDOS
          lxi     d,126
          dad     d           ; add 126 to Sys. Data Page addr.
          mov     e,m
          inx     h
          mov     d,m         ; DE = addr. of CTRL-P array
          push    d
          mvi     c,0a4h      ; Get List Number
          call    BDOS
          mov     e,a
          mvi     d,0
          pop     h
          dad     d
          mvi     m,0         ; reset CTRL-P flag
          ...                 ; console I/O echo is now off
```

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{\text{T.M.}}$ Operating System Release 2.0
Application Note 10, 9/14/81

COLD BOOT STARTUP

Applicable products and version numbers: MP/M II$^{\text{T.M.}}$ Release 2.0

MP/M II can execute one command upon cold boot. However, the system can execute any number of commands upon cold boot if the initial command is SUBMIT.

To execute the Startup command place the Startup command singularly into a file using standard command format. Name this file $n$.SUP where n is the console number that executes the command. The $n$.SUP file resides on the system drive at the desired USER number or at USER 0 with a SYS (SYSTEM) attribute. Examples are shown below.

Startup file:	0.SUP
Command in the Startup file:	SUBMIT START0

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{T.M.}$ Operating System Release 2.0
Application Note 11, 9/14/81

SUBMIT ENHANCEMENTS

Applicable products and version numbers: MP/M II$^{T.M.}$ Release 2.0

Enhancements to SUBMIT include the following new features and facilities.

INCREASED n.SUB FILE SIZE: SUBMIT file size is now unlimited. The n.SUB file originally was limited to one extent, 128 lines.

CHANGING THE USER NUMBER: To change the current USER number in SUBMIT, include the USER command in the SUBMIT file.

INCLUDE FILES: An include file is a standard SUBMIT file subject to all SUBMIT rules and features. Format for the INCLUDE command is demonstrated below.

```
$INCLUDE filename parm1 parm2 parm3 ...
```

The filename in the INCLUDE command must have the filetype .SUB to indicate a SUBMIT file and parameters are standard SUBMIT parameters. An INCLUDE file can nest up to four SUBMITs in a SUBMIT command.

EMBEDDED CONTROL CHARACTERS: Control characters can be embedded in a SUBMIT file by preceding the capitalized character with an ASCII up arrow ^. For example, type ^X to embed a CTRL-X. Embedded control characters are not interpreted by MP/M II, but can be of use to programs that SUBMIT executes.

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II T.M. Operating System Release 2.0
Application Note 12, 9/14/81

SPOOL UTILITY MODIFICATIONS

<u>Applicable products and version numbers:</u> MP/M II T.M. Release 2.0

SPOOL can return an error message if the file to be spooled is not found. To utilize this modification, SPOOL is divided into a transient portion (SPOOL.PRL) and a resident portion. The transient portion parses the command tail, opens the file, passes the file to the spool queue (named SPOOLQ), and displays an error message if the open sequence on the file fails. Then, the transient portion ends itself.

Issue a SPOOLQ command if you do not want to use a memory segment to spool a file. Error messages are not returned, however. Sample commands to spool a file are shown below.

```
SPOOL file1.typ,file2.typ ...
```

The SPOOL process passes the command tail, checks for errors, and sends the file to the spool queue (SPOOLQ).

```
SPOOLQ file1.typ,file2.typ ...
```

The command tail is sent to the spool queue (SPOOLQ) bypassing error checking or error reporting.

The SPOOL utility sets its priority to 201. Most processes execute ahead of the SPOOLER. To change the SPOOLER priority, the SPOOL.BRS file is modified. Make sure you have a back-up copy of SPOOL.BRS before using RDT to make the following changes.

```
A>rdt spool.brs
   ...
-s3b5
03B5 C9 c8
03B6 00 .
-ispool.brs
-w14
-g0
A>gensys
```

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{\text{T.M.}}$ Operating System Release 2.0
Application Note 13, 9/14/81

RECORD LOCKING/UNLOCKING

Applicable products and version numbers: MP/M II$^{\text{T.M.}}$ Release 2.0

Record locking/unlocking allows multiple processes to share access of one file. Files are opened in the UNLOCKED mode. A record locked by one process can only be read by a different process, however, a locked record can be modified by the initial process. Avoid reading locked records to prevent reading data that is being updated. To avoid reading locked records let the process try to lock the record. If the attempt fails, do not read the record. The following code segment demonstrates how to lock records.

```
        mvi     c,2ch       ;set multi-sector cnt.
        mvi     e,#         ;# = num. of sectors
        call    bdos        ;1<= # <=16
        mvi     c,2ah       ;lock record
        lxi     d,fcb       ;record to be locked
        call    bdos
```

The following code segment demonstrates how to unlock records.

```
        mvi     c,2ch       ;set multi-sector cnt.
        mvi     e,#         ;# = num. of sectors
        call    bdos        ;1<= # <=16
        mvi     c,2bh       ;unlock record
        lxi     d,fcb       ;record to be unlocked
        call    bdos
         ...
  fcb:
        db      0,'DATA',A0H,20H,20H,20H,'DAT',0
        ds      20
        db      10,0,0    ;begining at record 10
```

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II T.M. Operating System Release 2.0
Application Note 14, 9/14/81

GENSYS ENHANCEMENTS

Applicable products and version numbers: MP/M II T.M. Release 2.0

Enhancements to GENSYS include the following new features and facilities.

AUTOMATIC RESIDENT SYSTEM PROCESS INCLUSION FACILITY: The GENSYS automatic system generation facility can be modified to include all default disk .RSP files. Type GENSYS $AR to include the .RSP files automatically. The R option must be used in conjunction with the A option. Change the filetypes for files that you want to exclude from GENSYS.

ERROR RECOVERY: If an error is encountered running in automatic mode ($A option), GENSYS restarts in manual mode.

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{T.M.}$ Operating System Release 2.0
Application Note 15, 12/1/81

CHANGING THE PRIORITY OF SPOOL.PRL

Applicable products and version numbers: MP/M II$^{T.M.}$ Release 2.0

The SPOOL utility sets its priority to 201, therefore, most other processes execute ahead of the SPOOLER. Modify the SPOOL.PRL file to change the SPOOLER priority. If your product serial number is between 4-000-00001 and 4-000-00464, install MP/M II Patch 11 before changing the SPOOL.PRL default priority.

Make sure you have a back-up copy of SPOOL.PRL before using DDT to make the following changes.

```
A>ren spool.sav=spool.prl
A>ddt spool.sav
[MP/M II] DDT VERS 2.0
NEXT PC
0980 0100
-s269
0269 C9 c8
026A CD .
-ispool.prl
-w11
-g0

A>
```

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.

MP/M II$^{\text{T.M.}}$ Operating System Release 2.0
Application Note 16, 12/1/81

CHANGING THE BACKSPACE AND RUBOUT KEY FUNCTIONS

Applicable products and version numbers: MP/M II$^{\text{T.M.}}$ Release 2.0

Program: RESBDOS

Under MP/M II, the BACKSPACE key or CTRL-H (ASCII 08H) does a destructive backspace deleting the last character in the command buffer. The RUBOUT key (ASCII 7FH) or DELETE key deletes the last character in the command buffer and echoes it to the screen.

Procedure to reverse the BACKSPACE and RUBOUT key functions:

```
A>ren resbdos.sav=resbdos.spr
A>ddt resbdos.sav
[MP/M II] DDT VERS 2.0
NEXT  PC
0f80 0100
-sc0a
0C0A 08 7f
0C0B C2 .
-sc22
0C22 7F 08
0C23 C2 .
-iresbdos.spr
-wld
-g0

0A>gensys
```

Procedure to make RUBOUT identical to BACKSPACE:

```
A>ren resbdos.sav=resbdos.spr
A>ddt resbdos.sav
[MP/M II] DDT VERS 2.0
NEXT  PC
0f80 0100
-lc26
0C26  MOV  A,B
0C27  ORA  A
0C28  JZ   09F6
-ac26
0C26  jmp a0e
0C29  .
-bb28,1
-iresbdos.spr
-wld
-g0

0A>gensys
```

Procedure to make BACKSPACE identical to RUBOUT:

```
          A>ren resbdos.sav=resbdos.spr
          A>ddt resbdos.sav
          [MP/M II] DDT VERS 2.0
          NEXT  PC
          0f80 0100
          -lc0e
          0C0E  MOV  A,B
          0C0F  ORA  A
          0C10  JZ   09F6
          -ac0e
          0C0E  jmp a26
          0C11  .
          -bb10,1
          -iresbdos.spr
          -wld
          -g0

          0A>gensys
```

Licensed users are granted the right to include these enhancements in MP/M II Release 2.0 software.