# MP/M II
# Operating System
# USER'S GUIDE



Digital Research
P.O. Box 579
801 Lighthouse Avenue
Pacific Grove, CA 93950
(408) 6493896
TWX 910 360 5001

The MP/M II User's Guide was prepared using the Digital Research TEX80 Text Formatter and printed in the United States of America by Commercial Press/Monterey.

First Edition: August 1981 *
Second Edition: August 1982 *

# Foreword

MP/M II'' is an acronym for MultiProgramming Monitor Control **Program for Microprocessors.** It is & multiuser operating system for an eight bit microcomputer. MP/M II supports multiprogramming at each terminal. it is an upward compatible version of MP/M I.

a The MP/M II hardware environment must include an 8080, 8085 or Z80 microprocessor, at least 43K of random access memory (RAM) , a clock/timer interrupt, a floppy disk drive, and a console. A reasonable hardware configuration might consist of a hard disk and one floppy disk or other backup storage medium, two consoles and a printer.

MP/M II supports from one to sixteen logical or physical disk drives containing up to 512 megabytes of storage each, from one to sixteen printers, and from one to sixteen character 1/0 devices. of these character 1/0 devices, eight may be system consoles. A system console is a device such as a CRT terminal or teletype from which programs can be initiated. MP/M if supports up to 400 kilobytes of banked random access memory (RAM), and requires about 26 kilobytes of memory for itself.

Digital Research distributes MP/M II on two *standard format* IBM single density 8" floppy disks. The interface between the hardware and the software must be configured according to the instructions in the MP/M 11 System Guide. (That is, the MPMLDR and XIOS files must be customized for the target hardware, and the GENSYS program described in the System Guide must be used to generate an MPM.SYS file before MP/M II can be executed).

Properly written CP/M compatible programs run under MP/M 11 with little or no modification. The MP/M II Programmer's Guide provides the information needed to write MP/M II compatible programs.

The MP/M II User's Guide assumes that your MP/M II system is up and running. it contains the information you need to use the MP/M II operating system and to run applications programs under MP/M II.

The information in the MP/M II User's Guide is organized according to the anticipated order of need. Section 1 describes the initial console messages that appear on the screen after the system is brought up. Section 2 describes how to enter an MP/M II command and includes e. brief command summary. Section 3 is a complete description of MP/M II file specifications and related matters, including a discussion of how MP/M II searches for files. Section 4 describes the format arid conventions of command line syntax and examples in this manual, and points to the names and section numbers of the MP/M II utilities included on the MP/M II distribution disk. Sections 5 through 13 explain the function and use of those utilities.

The Appendixes offer brief encapsulated collections of MP/M II control characters, commands, options, error messages, and troubleshooting suggestions. Appendix A supplies an ASCII and Hexadecimal conversion table. Appendix B is a reference for common MP/M II file types. Appendix C summarizes MP/M !I control characters including command line editing controls and control character commands. Appendix D describes MP/M II error messages. Appendix E provides a troubleshooting checklist for locating files. Appendix F is a brief summary of the MP/M I' commands with some examples. Appendix G compares CP/M and MP/M commands. Appendix H summarizes the MP/M II commands "hat display disk and file status.

Even if you are familiar with CP/M or MP/M I commands, you should still read the introductory sections of this manual, tile new SDIR and SHOW commands in Section 6, and the SET command in Section 7. Most of the utilities have been enhanced.

# Table of Contents

# Table of Contents
# (continued)

## Table of Contents (continued)

# Table of Contents (continued)

# Appendixes

# Section 1
# MP/M II Sign-On Messages

## 1.1 MP/M II System Generation

Section 1 describes the messages that appear on the system consoles after bringing up the MP/M II system. Information in the messages is directly related to system generation, the process in which MP/M II is first brought up on a certain hardware configuration. System generation is covered in detail in the MP/M II System Guide. This section discusses the elements of system generation that affect MP/M II's sign-on messages.

over thirty utilities are supplied with MP/M II. Some of them can be incorporated into MP/M II at system generation time, or executed as separate command files from disk. Figure 11 is a directory display showing each utility with its disk space requirement in bytes.

```
OD>SDIR *.prl, *.com [size]

Directory For Drive D: User 0

D:   ABORT     PRL     1k  : ASM       PRL    10k  : CONSOLE    PRL     1k
D:   DDT       COM     6k  : DIR       PRL     2k  : DSKRESET   PRL     1k
D:   DUMP      PRL     1k  : ED        PRL     9k  : ERA        PRL     2k
D:   ERAQ      PRL     4k  : GENHEX    COM     1k  : GENMOD     COM     2k
D:   GENSYS    COM     9k  : LOAD      COM     2k  : MPMLDR     COM     6k
D:   MPMSTAT   PRL     4k  : PIP       PRL    10k  : PRINTER    PRL     1k
D:   PRLCOM    PRL     3k  : RDT       PRL     7k  : REN        PRL     3k
D:   SCHED     PRL     3k  : SDIR      PRL    18k  : SET        PRL     8k
D:   SHOW      PRL     8k  : SPOOL     PRL     2k  : STAT       PRL    10k
D:   STOPSPLR  PRL     1k  : SUBMIT    PRL     5k  : TOD        PRL     3k
D:   TYPE      PRL     2k  : USER      PRL     1k

Total Bytes:      = 146k    Total Records = 1059      Files Found     32
Total 1k Blocks   = 146     Used/Max Dir Entries For Drive D:      45/64
```

**Figure 11. Directory Display with Size of Files**

Besides specifying which utilities are to become part of MP/M II, system generation also specifies many system parameters. Some of the system generation parameters arc, listed below.

- the size and configuration of memory
- the number of consoles
- the number of printers
- which drive will be the system drive, the one on which MP/M II looks for files if they are not found on the default drive.
- which drive will contain any temporary files generated by the system
- the maximum number of locked records
- the maximum number of locked records per process
- the maximum number of open files
- the maximum number of open files per process
- which utilities will be incorporated into the MP/M II system as Resident System Processes (RSP) or Banked Resident System Processes (BRS), which are always accessible even though they are not present on disk as program files
- whether or not the day-file option is enabled to display the current time, as well as the drive and user area from which a program is loaded.

## 1.2 MP/M II Bootstrap Displays

After system generation, when MP/M II is first transferred or "booted" into memory, a system status display appears on console zero. A short boot message appears on all other consoles. Figure 12 shows a sample console zero MP/M II boot message, followed by the boot message that appears on all other consoles.

The system used in the figure was configured for an Altos microcomputer with an eight inch Winchester hard disk and an eight inch floppy disk drive. In this case, the floppy disk is Drive A and the hard disk is Drive E. Drive E has been designated as the system drive. The system drive is the one on which MP/M II looks for a specified file if it is not found on the default drive. This particular configuration supports two consoles and four memory banks as shown in the figure.

MP/M I I V2.0 Loader
Copyright (c) 1981 Digital Research
Number of Consoles = 2
Breakpoint RST = 6
Z80 alternate register set saved/restored by dispatcher
Memory Segment Table:

| | | | |
|---|---|---|---|
| SYSTEM | DAT | FFOOH | 0100H |
| TMPD | DAT | FEOOH | 0100H |
| USERSYS | STK | FDOOH | 0100H |
| XIOSJMP | TBL | FCOOH | 0100H |
| RESBDOS | SPR | FOOOH | OCOOH |
| XDOS | SPR | CEOOH | 2200H |
| BNKXIOS | SPR | BBOOH | 1300H |
| BNKBDOS | SPR | 9800H | 2300H |
| BNKXDOS | SPR | 9500H | 0300h |
| TMP | SPR | 9200H | 0300H |
| LCKLSTS | DAT | 8FOOH | 0300H |
| CONSOLE | DAT | 8DOOH | 0200H |

| | | | | | |
|---|---|---|---|---|---|
| MP/M II | Sys | 8DOOH | 7300H | Bank | 0 |
| Memseg | Usr | OOOOH | COOOH | Bank | 1 |
| Memseg | Usr | OOOOH | COOOH | Bank | 2 |
| Memseg | Usr | OOOOH | COOOH | Bank | 3 |
| Memseg | Usr | OOOOH | 8DOOH | Bank | 0 |

**Figure 11. Sample Boot Message for Console 0**

The following boot message appears on all consoles.

MP/M II V2.0
Copyright (c) 1981, Digital Research

## 1.3 The System Prompt

The boot messages are followed by the MP/M II system prompt. The prompt consists of a number, an alpha character, and a right angle bracket or greater than symbol, >. For example:

5A>

The first character of the prompt is a number from zero to f if teen. This number is the current or default user number. The user number indicates a unique region on the disk. Files are marked with the user number in which they reside. Therefore, it is not necessary to pre-allocate disk space to each user. No disk space is wasted if some user numbers are unused. Normally, you access only the files stored in this user number. The files in your current user number can be displayed by typing the MP/M II command "DIR". If you change the user number with the USER command described in Section 5, the number in the system prompt changes to reflect the new user number selected.

The second character of the MP/M II prompt is an alphabetic character which indicates the default drive. The default drive is the drive into which MP/M II is currently logged. It is the drive on which MP/M II first looks for a command file if a particular drive is not specified in the file specification. After a cold boot, the default drive specification is always A. You can change the default drive by typing the letter of the desired drive and a colon, followed by a carriage return as shown below.

```
5A>E:
5E>
```

Each console on a MP/M II system has a unique console number. The number of the console on which the main boot message appears is always zero, and additional consoles on the system are numbered console one, console two, and so forth. When the MP/M II system is initially booted, each console is assigned a different user number. Arbitrarily, the initial user number is the same as the console number. However, the console number has no relationship to the user number. The user number can be changed at any time, but the console number is not usually changed. Two independent users on the system can be in the same user number, but will riot normally be using the same console.

Table 11 shows typical console displays of the system prompt immediately after startup when the default drive is A. It also shows the system prompts after the default drive has been changed from A to C.

**Table 11. Sample System Prompts**

| | User 0 | User 1 | User 2 |
|---|---|---|---|
| Drive A | OA> | 1A> | 2A> |
| Drive C | OC> | lC> | 2C> |

## 1.4 The Day-File Option

The day-file option enables the display of the current time, as well as the drive and user area from which a program is loaded. The display appears just after a command is entered at the console. Figure 13 shows a listing of a sample distribution disk directory in which the day-file option has been enabled. Use the TOD command described in Section 10 of this manual to set the date and time correctly.

```
3D>DIR
15:53:32 D:DIR         PRL    (User 0)
Directory for User     3:
D: BNKBDOS   SPR : BNKXDOS   SPR : RESBDOS   SPR : TMP       SPR
D: XDOS      SPR : ABORT     RSP : MPMSTAT   RSP : SCHED     RSP
D: SPOOL     RSP : MPMSTAT   BRS : SCHED     BRS : SPOOL     BRS
D: ABORT     PRL : ASM       PRL : CONSOLE   PRL : DIR       PRL
D: DSKRESET  PRL : DUMP      PRL : ED        PRL : ERA       PRL
D: ERAQ      PRL : MPMSTAT   PRL : PIP       PRL : PRINTER   PRL
D: PRLCOM    PRL : RDT       PRL : REN       PRL : SCHED     PRL
D: SDIR      PRL : SET       PRL : SHOW      PRL : SPOOL     PRL
D: STAT      PRL : STOPSPLR  PRL : SUBMIT    PRL : TOD       PRL
D: TYPE      PRL : USER      PRL : DDT       COM : GENHEX    COM
D: GENMOD    COM : GENSYS    COM : LOAD      COM : MPMLDR    COM
```

**Figure 13. Sample MP/M II Directory**

The display just under the command line in Figure 13 is enabled by the day-file option. When the day-file option is enabled, it returns the time, the drive from which the program was loaded, the command program name and file type, and the user number from which it was accessed if it is other than the default user number. In this case, MP/M II shows the time as 15 hours, 53 minutes and 32 seconds (15:53:32) , using a twenty four hour clock. MP/M II found the DIR.PRL program on Drive D in user area 0.

When a program is resident in the system, such as a Resident System Process or, RSP file or is built into the system, as is tile ATTACH program, there is no actual physical file on the disk. Therefore, MP/M II cannot report a command program name and type. Instead, MP/M II returns the message:

```
15:55:10 Msq Qued
```

in the day-file option display.

# Section 2
# Introduction to MP/M II Commands

### 2.1 MP/M II Command Format

All MP/M II commands have the same basic parts and recognize the same filenames. The following sections explain in more detail MP/M 1I command format, MP/M II file specifications, and finally the actual use of the MP/M II commands.

In general, an MP/M II command line has three parts: the command keyword, the command tail, and a carriage return. in MP/M II, the command keyword must be typed next to the system prompt on the console. In the example below, TYPE is the command keyword arid B:DOCUMENT.LAW is the command tail.

**OA>TYPE B:DOCUMENT.LAW**

A command keyword identifies a program to be executed. The command keyword can be the name of a queue associated with an RSP, (Resident System Process) or it can be a command filename that identifies a program to be loaded from the default, or system, or, specified drive. The command tail can include simply a ("rive specification, or D file specification (see Section 3), and/or a list of one or more utility parameters or options. Sometimes the command tail is optional. in some cases, an absent command tail is "understood" as indicating the default drive, or all the files in the default user area on the default drive.

All commands MUST end with a carriage return keystroke, which signals the operating system to process the command. This means MP/M II can process only one command per line. To execute a sequence of commands, use the SUBMIT command described in Section 10. If you recognize a typing error or other mistake in your command before pressing the carriage return key, you can correct the error with the line editing controls shown in Table 2-1, below. The ^ character represents the CONTROL key on the keyboard. To enter a control keystroke, depress the CONTROL key and hold it down while depressing the desired alpha character.

MP/M II User's Guide 2.1 MP/M II Command Format

**Table 2-1. Command Line Editing Controls**

| Keystroke | Action |
|---|---|
| RUB | deletes character to the left of cursor, echoes character deleted cursor moves right |
| DEL | same as RUB |
| BACKSPACE | moves cursor back one space; erases previous character |
| ^H | same as BACKSPACE |
| ^U | cancels line, displays '#', cursor moves down one line and awaits a new command |
| ^X | deletes all characters in command line |
| ^R | retypes a "clean" line; useful after using RUE or DEL key |
| ^E | forces a physical carriage return, but does not send the command to MP/M II |
| RETURN | carriage return |
| ^M | same as carriage return |
| ^J | line feed, same as carriage return |
| ^Z | end of file, string or field separator |

MP/M II puts a few restrictions on command line length, but no restrictions on command letter case. The system internally translates all lowercase letters to uppercase, so you can enter MP/M II commands in either upper or lowercase, or a combination of both. MP/M II command lines can be as long as 100 characters. Your command is not sent to MP/M II until you press the carriage return key or until your command line length exceeds 100 characters.

## 2.2 MP/M II Command Summary

There are over thirty utilities supplied with MP/M II. Each utility is invoked by typing its name (command filename) next to the system prompt on the console. Table 22 provides a brief summary of the available MP/M 11 commands. Twenty of these utilities are described individually in Sections 5 through 13 of this manual. Programming utilities are described in the MP/M II Programer's Guide. System generation utilities are described in the MP/M II System Guide.

**Table 2-2. MP/M II Utilities**

| Name | Action |
|---|---|
| ABORT | Aborts a specified process |
| ATTACH | Attaches a program to its console (built in) |
| ASM | Assembler |
| CONSOLE | Displays console number |
| DDT | Dynamic debugging tool |
| DIR | Displays disk directory |
| DSKRESET | Resets drives |
| DUMP | Displays file in hexadecimal format |
| ED | Editor |
| ERA | Erases a file |
| ERAQ | Erases file with confirmation query |
| GENHEX | Converts COM file to HEX file |
| GENMOD | Converts HEX file to PRL file |
| GENSYS | Generates MP/M II operating system |
| LIB | Creates a library of relocatable REL files |
| LINK | Links a number of REL files together |
| LOAD | Converts HEX file to COM file |
| MPMSTAT | Displays MP/M II internal status |
| MPMLDR | Loads MP/M II operating system |
| PIP | Copies files |
| PRINTER | Displays and sets printer number |
| PRLCOM | Converts PRL file to COM file |
| RDT | Relocatable debugging tool |
| REN | Renames files |
| RMAC | A relocatable Macro Assembler |

**Table 2–2. (continued)**

| Name | Action |
|---|---|
| SCHED | Schedules processes for specified date and time |
| SDIR | Displays disk directory with options |
| SET | Sets file and disk protection levels, file time stamping, and file attributes. |
| SHOW | Shows disk status and protection levels |
| SPOOL | Spools files to the list device |
| STAT | Displays and sets file and disk status |
| STOPSPL | Aborts the spooler |
| SUBMIT | Submits a batch processing file |
| TOD | Displays and sets the time and date |
| TYPE | Di splays ACSII file contents at the console |
| USER | Displays and sets user number |

## 2.3 Control Character Commands

MP/M II has a set of control character commands that start and stop screen scrolling, echo console input at the printer, and detach and abort programs. Table 23 below summarizes these control character commands and their uses. As in Table 22, the T character indicates the CONTROL Key on the keyboard. To enter a control keystroke, hold the CONTROL key down while depressing the desired alpha character.

**Table 23. Control Character Commands**

| Keystroke | Action |
|---|---|
| ^P | echoes all console output to the printer; a second ^P ends printer echo. This only works if your system is connected to a printer. |
| ^s | stops console listing temporarily; ^Q resumes the listing. |
| ^Q | resumes console listing after ^S, otherwise it is ignored. |

**Table 2-3. (continued)**

| Keystroke | Action |
|---|---|
| ^C | prompts to abort a program currently running at a given console. |
| ^D | detaches the currently executing program from the console at which the ^D is entered. If no program is executing, the ^D reattaches detached programs waiting for the console. |

After you enter a TS to stop a console display, you have two options: you can enter ^Q to resume the display or you can enter ^C to abort the process. After a TS, MP/M II responds to any input character other than ^Q or ^C by sounding the console bell or beeper.

Note that some applications programs trap all the Control Characters for their own purposes. This is particularly true of word processing programs. For example, in a word processing application a ^C might cause a screen full of text to scroll by. In this case, the ^C does NOT abort the Word Processing Program. Exit the program using its own commands, or use the ABORT command from another console.

## 2.4 Attaching and Detaching Processes

MP/M II supports multiprogramming at each system console. You can initiate a program at a console and then detach the console from that program with the TD character. Then you can initiate another program at the same console. You can continue to initiate and detach programs until all of the system's existing memory segments have been allocated. Memory segments become free again as programs finish executing or are aborted with the ^C or ABORT command, described in Section 8.

To finish executing, programs that require console interaction must be reattached to the console using the ^D or ATTACH command. The ^D reattaches the programs in the same order in which the programs were detached. The ATTACH command attaches the program you specify, independently of the order in which the program was detached (see Section 8). ^C simply aborts the program and frees the memory space.

It is recommended that you experiment with the TD and ATTACH commands . Use the MPMSTAT command to display the status of various system functions to see in which memory segments the programs you have attached and detached are located.

# Section 3
# MP/M II Files

### 3.1 MP/M II File Specifications

A file is a collection of data stored on disk. A file is given a unique name and that name is used to access that file. Disk directories display a list of the filenames stored on the disk. A command file is an executable file, a series of instructions that the computer can follow step by step. A command file is generally referred to as a "program."

A command file sometimes requires a data file to process. A data file is generally a collection of data; a list of names and addresses, the inventory of a store, the accounting records of a business, a document , parts of a book , scientific weather information, or other collections of similar information. In a sense, a data file is the object of a command. Sometimes a data file is a program file, but in this case the program file is itself the object of another command file. This is the case when using a command to copy program files from one disk to another, for example.

There are three ways to create a file. You can create a file by copying an existing file to a new location, perhaps renaming it in the operation (refer to Section 12, PIP, MP/M II's Peripheral Interchange Program) . The second way to create a file is by using a text editor. The text editor creates the file and assigns the name you specify to the file (see Section 11, ED, the MP/M II Text Editor). Finally, some programs create output files.

MP/M II identifies every file by its unique file specification. A file specification can consist of f o u r p a r t s a drive specification, a primary filename, a file type, and a password as shown below.

```
d:filename.typ;password
```

A drive specification consists of ~, single letter drive name followed by a color.. Either a primary filename or a filetype must be present; the remaining fields are optional. if you do specify a filetype, it must be preceded by a period. Note that in the remainder of this document the general term "filename" refers to both the primary filename and the optional filetype. if you specify a password, you must separate it from the filename with a semicolon. Spaces are not allowed in MP/M 11 filenames.

Note: In the Syntax lines in the following sections of this User's Guide, the term "filespec" indicates any valid combination of the elements included in ,he file specification. That is, a drive specification, a primary filename, a filetype and a password. Valid combinations are:

- filename
- filename.typ
- d:filename
- d:filename.typ
- filename.typ;password
- d:filename.typ;password

A complete file specification with all possible elements included consists of a drive specification, a primary filename, a filetype, and a password, all separated by their appropriate delimiters, as shown below.

```
A:DOCUMENT.LAW;Secret
```

The following sections define each of the four parts of a file specification.

### 3.1.1 Drive Specifications

The drive specification, d:, designates the file's location. If the file is on your default drive, you need not enter a drive specification. "Default" indicates the current drive and the current user number. 'The default drive and default user number always match the drive and user number in the system prompt. These defaults are the drive and user number in which the system first searches for files if F., particular drive is not specified.

To designate a file not on your default drive, replace d with the letter name of the drive that contains the desired disk file. This part of the file specification changes when you move the disk containing the file to another disk drive.

### 3.1.2 Primary Filenames

The primary filename, which is usually provided by the user when the file is created, normally tells something about the contents of the file. A filename is from one to eight characters long, and can contain any letter or number. Some special characters are also allowed. Section 3.1.5 defines the special characters that are not allowed in file specifications.

### 3.1.3 Filetypes

Generally, a file specification includes a period and a three letter filetype. Like a primary filename, a filetype can contain any letter or number, but not the special characters listed in Section 3.1.5.

Normally the filetype tells something about the file. Some programs require that their input files be a certain filetype. For example, MP/M II requires that an executable command file be in a certain format and have the filetype .COM or .PRL. Not all programs require specific file types. For example, the MP/M II text editor, ED, accepts any filetype. For this kind of program, you can give the input file any filetype that seems convenient, or give it no filetype at all.

The user assigns the filetype to the file when he creates it. When a program manipulates a file, it might change the filetype to indicate that the file has been modified. For example, when ED finishes editing a file, it changes the original filetype to .BAK, then gives the new, edited file the original primary filename and filetype. See Appendix B for a list of MP/M II filetypes and their general meanings.

### 3.1.4 Passwords

MP/M II supports password protected files. Passwords are valuable in a multiuser system because they enable each user to protect his files from accidental or intentional damage by other users. Passwords enable managers and systems personnel to allow limited access to certain files for security purposes.

A password is an optional part of the file specification. it always appears next to the filename in a command line. It is separated from the filename by a semicolon. Consider the password as part of the file specification when entering drive specifications or options in command lines.

The **PASSWORD** option of the SET command (see Section 7) can assign a password to any file. This means that all executable programs, commands and data files can have password protection. Furthermore, the command files ED, ERA, ERAQ, PIP, REN and TYPE accomodate passwords in their data filename. This means that a command line can require two passwords to execute properly. The first password is needed to access the command program. The second password is necessary to access the file. specified in the command tail. In the following examples of command lines with passwords, assume that all files have been assigned the password "XYZ".

```
OA>TYPE;XYZ
OA>TYPE;XYZ B:CAT.ASM;XYZ
OA>REN;XYZ NEWNAME.TYP = OLDNAME.TYP;XYZ
OA>ED;XYZ DOCUMENT.LAW;XYZ
OA>ERA;XYZ C:*.*;XYZ
```

Some MP/M. II commands and most word processing, accounting packages and other applications programs running under MP/M II do not accept passwords in the command tail. If you wish to protect your files, set the default password before executing the application program. For example, you would not have to specify the password "XYZ" in the above examples if you first issued the following set default password command. (See the SET command described in Section 7 of this manual).

```
SET [DEFAULT = XYZ]
```

MP/M II displays the following message when a required password is missing or incorrect.

```
Bdos Err On d:        Password Error
Bdos Function: NNN    File:  FILENAME.TYP
```

Passwords can contain any characters except for those listed in Table 31. All passwords are converted to uppercase when entered in f ile specifications or in the standard MP/M II utilities. Application programs using the password protection features of MP/M II, however, may distinguish between upper and lowercase passwords.

### 3.1.5 Special Characters in File Specifications

The characters in Table 3-1 have special meaning in MP/M II command lines and should not be used in file specifications. All other special characters are allowed.

Table 31. Special Characters

| Character | Meaning |
|---|---|
| < = ,<br>tab space<br>carriage return | file specification delimiters |
| : | drive delimiter in file specification |
| . | file type delimiter in file specification |
| ; | password delimiter in file specification |
| * ? | wildcard characters in file specification |
| < > & | reserved for 1/0 redirection |
| [ ] | option list delimiters |

MP/M II User's Guide 3.1 MP/M II File Specifications

**Table 3-1. (continued)**

| Character | Meaning |
|---|---|
| ( ) | delimiters for multiple modifiers in option list |
| / $ | option delimiters |
| ; | comment delimiter in column one |

The less than, equal, comma, tab, space and Carriage return characters separate file references and other items in the command line. The colon and period delimit drive specifications and filetypes in file specifications, respectively. A semicolon within a file reference delimits a password. The asterisk and question mark characters, * and ?, are wildcard characters in ambiguous file specifications (see Section 3.3). The less than and greater than characters, < and >, are reserved for future use. Square brackets, [ and ], isolate an option or option list from its command keyword (global option) or from its file specification (local option). Parentheses, ( and ;, are used to isolate a list of more than one modifier , inside the square brackets, for options which have modifiers (see tile SDIR utility). The slash, /, and dollar sign, $, are reserved for the specification of options in the command line. A semicolon at the beginning of a command line indicates that the line is a comment.

### 3.2 Ambiguous File Specifications

The MP/M II commands can select and process several files when a special filename is included in the command tail. This special ambiguous filename can refer to more than one file because it gives MP/M II a pattern to match: MP/M II searches the disk directory and selects any file whose filename matches tile pattern. DIR, SDIR, ERA, ERAQ, STAT, PIP, REN and SET accept. an ambiguous filename in a file specification.

To make a filename ambiguous, replace characters in the filename or filetype with "wildcard" characters . The wildcard characters are ?, which matches any single letter in the same position, and *, which matches any characters in the rest of the filename or filetype. wildcard characters Eire not valid in passwords or drive specifications.

The wildcard characters can march certain parts of filenames. For example, to reference only the files with the primary filename PROG, use PROG.*. To reference only the files with the filetype BAK, use *.BAK in the command tail. To reference all files on the default drive or disk, use *.*. The reference APP?.TXT selects all of the following if they exist on the current disk:

| | | |
|---|---|---|
| APPA.TXT | APPI.TXT | APP3.TXT |
| APPB.TXT | APP2.TXT | APP.TXT |
| APPC.TXT | | |

but not these because they do not match the APP?.TXT pattern:

| | | |
|---|---|---|
| APP14.TXT | APP | APIP.TXT |
| FILEI.APP | APP2.TEX | |

The reference APP*.*, however, matches all filenames beginning with APP:

| | | |
|---|---|---|
| APPA.TXT | APPl.TXT | APIP.TXT |
| APPB.TXT | APP2.TXT | APP2.TEX |
| APPC.TXT | APP3.TXT | APP |
| APP14.TXT | APP.TXT | |

Commands that accept ambiguous filename also accept a drive specification as a part of the reference; however, wildcard characters are not allowed in a drive specification. For example,

**OA>DIR B:*.BAK**

is acceptable and lists all .BAK files residing on disk B. However,

**OA>DIR ?:*.BAK**

is illegal and result, in a "Bad entry" error message.

## 3.3 File Attributes

A file attribute is a characteristic that you can assign to a file. The attributes affect whether or not the file appears in normal directory displays (see Section 6, SDIR), whether or not the file can be accessed from other drives or user areas, and whether the file can only be read or both read and written to.

SET and STAT can assign two accessing attributes to files (se - Sections 6 and 7, the STAT and SET commands). The first attribute can be either DIR (Directory) or SYS (System). You can access a command file or a data file that has the DIR attribute only if the file is in the default user area of the default or specified drive. Remember the default user area and drive are those displayed in the MP/M II prompt. You can access a command file or a data file that has the SYS attribute, the fi1e is in the default user area or in user area 0 of the default or specified drive . You can also access

access a command file that has the SYS attribute if it is in the default or user 0 area of the system drive.

The other attribute SET and STAT can assign to a file is either RO (Read Only) or RW (Read Write) If a particular file is set to RO, an attempt to write data to that file produces a Read only error. A file with the RW attribute can be read or written to at any time unless it is password protected or the entire drive is set to Read Only.

MP/M 11 reads from any disk any time. However, if you change a disk and do riot reset the drive with the DSKRESET command, the entire drive become Read only and MP/M II will riot write to that disk. Therefore, it. is possible to lose an entire edit a file to which you are writing changes if you do not reset the drive when you change a disk.

A third file attribute, the archive attribute, is set by the PIP command with the [A] option. When You make a copy of a group of files using PIP with the [A] option, the files are marked ("archived") after copying is completed. The archive option in PIP only copies files which have not been already archived. it must be used with an ambiguous file specification. SDIR and STAT report archived files by listing an "A" in their file attributes column.

MP/M II also supports four user-defined file attributes labeled Fl , F2, F3, and F4. A SET command can set each of these attributes to either on or off . SDIR and STAT list the number (1-4) of any of the user-defined attributes which are set to on.

### 3.4 XFCB Information

MP/M II can record certain optional information about a file in an Extended File Control Block (XFCB). MP/M I I uses a. File Control Block (FCB) to help locate a file on disk. In the XFCB, MP/M II can record a, password and two time stamps for the file. One time stamp can record when the file was last updated. The other time stamp can record either when the file was created or last accessed. See the descriptions of the SDIR and SET commands in Sections 6 and 7 or E complete discussion of XFCBs, password creation, and time stamps..

### 3.5 File Location Conventions

This section describes the factors to consider when entering a command at your console, especially if it appears that MP/M 11 cannot find a specified file. MP/M II might answer your command line with a question mark even though you are sure your command file (executable program file) is on the disk. The command program might return E. "File not found" message even though you are also sure your data, file is on the disk. Remember that you have multiple drives and you might riot be. logged into the drive you need.

Remember also that there are sixteen available user areas on a single drive. Normally, a file cannot be accessed unless it is in the default (currently specified) user area. If an optional drive reference is specified, the file must be in the same user area on the optional drive as it is on the default drive.

The following two sections describe how MP/M 11 searches for command files and data files. The last section offers a list of troubleshooting suggestions to help you if it appears that MP/M II cannot find your file.

### 3.5.1 Command File Searches

When you enter a command, MP/M IT first checks its "message list" or queue list to see if the command is a Resident System Process (RSP) , and therefore resident in memory and not on disk. if MP/M IT does not find the specified command names a queue associated with an RSP, it looks for the specified command as a file on disk

MP/M II searches for two command filetypes in four separate locations before returning a question mark. The system first looks for a file of type PRL (page relocatable). If it does not find c~ PRL file, MP/M IT looks for a file of type COM (Command) . MP/M IT checks for both filetypes in four locations it necessary.

MP/M II first looks for a specified file of type PRL under the default user number on the default drive. For the second search MP/M II checks for the PRL file under user number 0 on the default drive. if that search fails, the system looks for a file of type COM under the default user number on the default drive , and then for a file of type COM under user 0 of the default drive. If that search fails too, MP/M II looks for the PRL file in a third location, under the default user number on the system drive. Next, MP/M II looks for the PRL file in a fourth location, under user number 0 of the system drive. if MP/M II does not find the PRL file it then looks for a fi.~ of type COM in the default user number of the system drive, and " Iken in user 0 of the system drive. MP/M II can only find the file in the second, third and fourth locations if it has the SYS attribute on. However, if you are in user 0 to begin with, MP/M 11 can finG your f ile on the default or specified drive if it has an attribute of DIR. If you are already on the system drive, MP/M IT can find your file on the system drive in your default or, specified user number if it has an attribute of DIR. If MP/M II does not find the file, it displays your requested filename followed by a question mark.

Table 3-2 shows the order in which MP/M II looks for a command file on the disk drives after it has checked for Resident System Processes.

**Table 3-2. Command File Search Order and Locations**

| Search Number | For Filetype | In User Number | On Drive |
|---|---|---|---|
| 1 | PRL | default | default or specified |
| 2 | PRL | 0 | default or specified |
| 3 | COM | default | default or Specified |
| 4 | COM | 0 | default or Specified |
| 5 | PRL | default | system |
| 6 | PRL | 0 | system |
| 7 | COM | default | system |
| 8 | COM | 0 | system |

There are cases when MP/M II does not perform all eight searches. of course when MP/M II finds the file, it searches no further. However, if the command file specification includes a drive specification the search pattern is changed.

If the command file specification includes a drivc specification, MP/M II looks for the file only on the specified drive. First MP/M II looks for a file of type PRL in the default user area and then in user 0 of the specified drive . Then MP/M II looks for a file of type COM in the default user area, and then in user 0 of the specified drive.

If the command file specification includes either a drive specification or a password, MP/M II automatically searches for the program file on a disk drive and does not check for RSPs. if the command identifies a, queue associated with an RSP, do not use a drive or password in referencing it.

### 3.5.2 Data File Searches

MP/M II checks for data files in two locations only. First it looks for the data file in the default user area on the default or specified drive. Then MP/M II looks for the data file in user area 0 of the default or specified drive. Unless the default user area is 0, MP/M II can only find the file in user area 0 if the file has a SYS attribute. Table 33 summarizes how MP/M II searches for data files.

**Table 33. Data File Search Locations**

| Search Number | In User Number | On Drive |
|---|---|---|
| 1 | default | default or specified |
| 2 | 0 | default or specified |

If the data file happens to be opened in unlocked mode, MP/M 11 does not look for it on the system drive. (Refer to Section 2 of the MP/M II Programmer ' s Guide for a complete discussion of the modes involved in opening and closing files. This is relevant to the number of users accessing a given file at one time, and whether or not a user is trying to write to that file) . If the command program does not f in' the data file, it generally displays the following message:

"File not found."

### 3.5.3 Troubleshooting File Searches

If you are having trouble, the following checklist should help you remember the factors involved when accessing files. This list is reproduced in Appendix E.

- If the drive is set to a different density than the disk inserted in it, MP/M Il returns a Bad Sector error. (See Appendix D. MP/M _7l Error Messages).

- If the file is set to Read Only, you can read the file but you cannot write to the file.

- If the drive is set to Read Only, you can read from files on the drive but you cannot write to them. This might happen if you have forgotten to use DSKRESET before changing your disk .

- If you have accidentally or otherwise typed a TS, your keyboard will be locked until a ^Q unlocks it.

If you receive a "reloc seg not free" message, use a ^D to reattach a process to the console so it can finish executing and free a memory segment. This situation could also occur if you accidentally typed a TD and didn't realize it.

Files with the DIR attribute can only be accessed if they are in the default user area on the default or specified drive.

Files with the SYS attribute can be accessed if they are in the default user area or user 0 of the default or specified drive.

If a drive is specified in the file specification, MP/M 11 only looks for the file in the default and zero user areas of the specified drive.

If the command line specifies a drive or a password end the command identifies a queue associated with Resident System Process, MP/M II will not find the command.

If the file is password protected, you might get a password error message.

Is the password protection mode set to READ, WRITE, DELETE or NONE? (SDIR displays the protection mode, see Section 6).

> If the password protection mode is set to READ, the n you need a password to read the file.
>
> If the password protection mode is set to WRITE, you can read the file without supplying the password, but you need the password to write to the file.
>
> If the password protection mode is set to DELETE, you can read or w r i t e to the file, but you need the password to erase it.
>
> if the mode is set to NONE, the password is erased; you no longer need it at all.

Does the drive label have a password assigned to it? (See the SET command in Section 7).

> If the drive label has a password and password protection is turned on for the drive, then you need a password to access any password protected files on that drive.

The simplest method of locating a file under MP/M II is to use the global search facilities of the SDIR command. Suppose you wish to locate the file TWO. TEX. You think it is somewhere on the system but you can' t seem to find it. The example below shows how SDIR can be used to locate a "lost" file.

```
1A>SDIR [USERS=ALL, DRIVES=ALL] TWO.TEX

Directory For Drive E: User 10

    Name      Bytes   Recs   Attributes    Prot     Update    Create

TWO     TEX     10k    75 Dir RW         Read

Directory For Drive M: User 9

    Name      Bytes Recs Attributes

TWO     TEX     10k    75 Sys RW
```

The above SDIR command located the file in two places: on user 10 of drive E and on user 9 of drive M. On drive E, the file is password protected with a protection level of Read. On drive M, it is not protected but does have the System attribute which causes it to be missing from normal directory displays. The shorter display for drive M means that there is no directory label for that drive

# Section 4
# Introduction to Utility Programs

## 4.1 Organization of Utilities

The utilities are grouped in the order you will need them, according to common characteristics, in order of importance to MP/M II and with the new ones in or near the beginning.

Section 5 begins the discussion of the MP/M II utilities with the DSKRESET, USER and CONSOLE commands. Section 7 introduces the SET command. Section 6 discusses the DIR and STAT utilities, and introduces the SDIR and SHOW commands. Section 8 continues with the MPMSTAT, ATTACH and ABORT utilities. Section 9 groups the TYPE, REN, ERA and ERAQ commands. Section 10 describes the TOD, SCHED, PRINTER, SPOOL, STOPSPLR and SUBMIT utilities. Section 11 is devoted to PIP, the Peripheral Interchange Program. Section 12 is devoted exclusively to MP/M II's text editor, ED.

## 4.2 Conventions and Nomenclature

The conventions for command line syntax in the descriptions of each utility are listed below.

When there are several ways to enter a given command, each way is shown on a separate line. The minimum command is shown first, followed by longer variations of the command, and finally by any optional items in the command line.

You can always replace "d:" with a drive specification.

You can always replace "n" with a number.

You can always replace "filename" with an actual filename.

You can always replace "typ" with an actual filetype.

When "typ" is riot specified in the syntax example, no filetype is necessary.

The term "filespec" indicates any valid combination of file specification elements; the drive, the filename, the filetype, and/or the password (see Section 3.1).

You can always replace "programname" with the filename of an executable program.

Special symbols represent the control key and the carriage return key in some examples in this manual. Do not enter these symbols as part of an MP/M II command. The special symbols ate defined below.

An up-arrow indicates that you should enter a "control" keystroke; hold down the control key and strike the desired character. The up-arrow appears only in descriptions in the text and in examples showing input. When you enter a control character, most terminals show the output character on the screen preceeded by a circumflex, (for example, ^C.) Therefore, the examples showing a screen output display in the following sections use a circumflex to show when a control character has been entered.

<cr> This symbol represents a carriage return keystroke. This symbol appears only when an example needs clarification; f o r example when the user's only input is to press the carriage return key. Remember, however, that every MP/M II command must be terminated by a carriage return keystroke, whether or not <cr> appears in the example.

To clarify examples of interactions between the user and the operating system, the characters entered by the user are shown in **boldface**. MP/M II's responses are shown in normal type.

## 4.3 Options in Utility Command Lines

MP/M II supports two kinds of options in the command line. A global option applies to the entire command line and is placed in square brackets just after the command keyword. A local option applies only to a specific file, and is placed in square brackets just after the filename or filename and filetype to which it applies.

An option can sometimes have a modifier. In the following examples, "Pass" is the option and "secret" is the modifier. "Drive" is the option and "B" is the modifier.

```
SET [PASS = Secret]
SDIR [drive = B]
```

Sometimes a modifier can include more than one element. In this case, the modifier is enclosed in parentheses. The examples below illustrate options with a modifier of more than one element

```
SDIR [drive    (a,b,c,d)]
SDIR [user    (0,1,2), SIZE]
```

In the above examples, "drive," "user" and "size" are tile options. The modifiers are (a,b,c,d) and (0,1,2).

Generally, options can be strung together and separated by commas or spaces within one set of brackets. You only need to specify one or two letters of the option keyword to identify the option.

The examples below illustrate the global option. it is placed just after the command keyword and applies to the command keyword. SDIR is a utility that has global options.

```
SDIR [short]
SDIR [drive = all, user = all] *.PRL
SDIR [SYS, DIR, Drive = (a,b,c)] document.law
```

The following examples illustrate local options. Local options are placed just after the filename to which they apply.

```
SET A:DOCUMENT.LAW [PASS = Secret]
SET B:*.*[RO, SYS]
PIP B:GENLEDR.DAT [G3] = A:GENLEDGR.WRK [V,GO]
PIP B: = A:DOCUMENT.LAW;Secret [VJ
```

# Section 5
# DSKRESET, USER, CONSOLE

### 5.1 The DSKRESET Command

**Syntax:**

```
DSKRESET
DSKRESET d:
DSKRESET d:,d:,d: ...
```

The DSKRESET command enables the operator to change disks. The. DSKRESET command with no command tail resets all the disk drives. DSKRESET with specified drives in the command tail resets only those drives. After turning on the system, MP/M II automatically resets all of the drives. It is important to reset a drive with the DSKRESET command before changing the disk in that drive. If the disk reset is successful, it means no one else is using the disk and it can be safely changed.

The DSKRESET command checks the drive for any open files. if DSKRESET doesn't find any open files, it resets the drive. The following exchange demonstrates what happens if DSKRESET finds an open file on a drive.

**OA>DSKRESET B:**
Disk reset denied, Drive B: Console 2 Program PIP

MP/M II has denied the request to reset the disk in drive B because drive B has an open file on it, namely the program PIP, which was initiated at console 2.

When changing disks, you must reset the drive to enable writing to the new disk. If you change a disk and forget to reset the drive and then try to write data to that disk, MP/M II will notice that the drive has not been reset, automatically set that drive to Read only, and will not write data to any files on that disk. Therefore, if you forget to reset the drive when you change a disk, it is possible to lose an entire edit (changes you are making to a file).

Note: It is extremely important to execute the DSKRESET command BEFORE changing a disk, in case another user has open files on that disk. If a disk is removed from a drive while a process is executing, the integrity of the data in the open file on that disk might be irrevocably damaged.

## 5.2 The USER Command

**Syntax:**

```
USER
USER n
```

The USER command has two functions. When entered without a command tail, it displays the current (default) user number. This number is always the same as the number to the left of the drive specifier in the system prompt, as illustrated below.

```
OA>USER
User Number = 0
OA>

5E>USER
User Number = 5
5E>
```

The USER command entered with an argument n changes the current user area to the number specified by n. n can be any number from zero to fifteen. The examples below illustrate how to change user numbers.

```
OB>USER 3
User Number
3B>

2F>USER 1
User Number
IF>
```

MP/M II uses the identification or user number selected by a USER command to control file access. It assigns the selected user number to each file created in that user area. For example, if a file was created by user 8, MP/M II assigns the number 8 to that file, and generally allows only user 8 to access it later.

MP/M II supports up to sixteen users numbered 0 to 15. In response to a request for a directory listing, MP/M II displays only those files that match the current user number. In effect, the USER feature allows each user to access only the area of the directory that contains his files. Note that each user can still rcference every disk drive in the system, but that user has access only to files created under his user number. However, if a file is assigned an attribute of SYS by a SET or STAT command, it becomes a "system" file and is accessible to other users. This means that utilities and systems programs can be available to everyone on the system. In addition to the default user number, the SYS attribute allows a file to be accessed if it is in User 0 of the default drive, the default user area of the system drive, or in User 0 of the system drive.

Only versions 2.0 and greater of CP/M and all versions of MP/M can create or access files on user numbers other than user 0. However, a file created with an older version of CP/M can still be accessed by MP/M II. Because such a file was not assigned a user number when it was created, it in effect has a user number of 0.

Most MP/M II commands access only the current or default user's area of the directory. An ERA *.* erases only the current user's files. DIR lists only those files stored in the user's area, REN can change only those filenames, and TYPE can display only those files. The PIP command is an exception to this rule. PIP can copy a file from or to a different user area. Use the G option in PIP to specify the user number (see Section 11, The PIP Command).

## 5.3 The CONSOLE Command

Syntax:

```
CONSOLE
```

Each console attached to an MP/M II system is assigned a console number at system generation time. The console numbers range from zero to fifteen. The console at which the long boot message appears is Console 0. Other system consoles can be numbered I to 7. On bootstrap, the user number in the system prompt is arbitrarily set to match the console number. For example, on bootstrap console 3 is assigned user number 3. You do not usually need to know what your console number is, although it may be displayed in certain error messages, such as the DSKRESET denied message described in Section 5.1. The ABORT and STOPSPLR commands described in Sections 8.3 and 10.5 use the console number. The user number is very useful and is displayed in the MP/M II prompt along with the default drive.

To learn your console number, use the CONSOLE command. The CONSOLE command returns the number of the console at which the command was entered. The example below illustrates a possible exchange using the CONSOLE command.

```
OA>CONSOLE
Console = 0
OA>USER 3
User Number ::= 3
3A>CONSOLE
Console = 0
3A>
```

# Section 6
# DIR, SDIR, STAT, SHOW

## 6.1 The DIR Command

**Syntax:**

```
DIR
DIR d:
DIR filespec
DIR filespec, filespec
DIR filespec [SYS]
DIR filespec [Gn]
```

DIR displays the filenames stored on a disk directory. DIR displays only the filenames assigned to the default user area of the disk directory. Remember that the default user area is indicated by the number, 0 to 15, that appears to the left of the drive letter (A,B,C,D,E,F .... P) in the system prompt. DIR alone does not display files with the system attribute, (SYS) . Use DIR with the SYS option to display files with the SYS attribute turned on.

DIR needs no command tail, but accepts either a drive specification or a filename or both. The filename can use the * and ? to indicate wildcards. In response to DIR without a command tail, MP/M II lists the filenames on the default drive, in the default
user area, as shown below:

```
OA>DIR
Directory for User 0:
A: DIR       PRL : PIP       PRL : SUBMIT    PRL : ERAQ      PRL
A: ED        PRL : ASM       PRL : DDT       PRL : LOAD      PRL
A: STAT      PRL : TOD       PRL : DUMP      PRL : DUMP      ASM
A: TEST1     ASM : TEST2     ASM : DEBLOCK   ASM : DISKDEF   LIB
```

If no command tail is given, as in the above example, MP/M II treats the command as if an ambiguous *.* filename were included and displays all the filenames on the default drive in the default user area. DIR with a drive specification is also treated as if *.* were entered, but displays the directory of the requested drive, still in the same user area, as shown in the sequence below:

```
OA>DIR B:
Directory for User 0:
B: DOCFILEl TXT : DOCFILE2 TXT : DOCFILE3 TXT : NEWPROG ASM
B: NEWPROG BAK  : ADDPROG ASM  : DOCFILEl BAK : CHECKPRG ASM
B: DOCFILE3 BAK : CHECKPRG BAK : ADDPROG BAK  : DOCFILE2 BAK
```

DIR with a specific file specification searches the directory for the requested filename and displays it if it exists. Otherwise, the system returns a "File not found" error message. DIR with art ambiguous filename displays any filename in the directory that matches, as the examples below show.

```
OA>DIR *.PRL
Directory for User 0:
A: DIR      PRL : PIP      PRL : SUBMIT   PRL : ERAQ     PRL
A: ED       PRL : ASM      PRL : DDT      PRL : LOAD     PRL
A: STAT     PRL : TOD      PRL : DUMP     PRL
```

```
OA>DIR B:DOCFILE?.*
Directory for User 0:
B: DOCFILEl TXT : DOCFILE2 TXT : DOCFILE3 TXT : DOCFILEI BAK
B: DOCFILE3 BAK : DOCFILE2 BAK
```

Use DIR not only to check the contents of your disk, but also to verify that any file operations such as renaming, erasing or moving have been performed correctly. If your disk directory is longer than your console screen can display at one time, you might need to type a TS to temporarily halt the console display before the top scrolls past you. To continue the display, type a TQ If you strike any other key during the directory display, DIR aborts immediately.

You might want to make a printed copy of a disk directory to keep with the disk. To do this, enter a TP before entering the DIR command. MP/M II then lists the directory at the printer as well as the console. Enter another ^P to stop the echoing of all console activity at the printer.

**OA>DIR *.PRL [SYS]**

The [SYS] option shown in the example above causes DIR to display any system files residing on the drive. These files are normally invisible to the DIR command.

The System file attribute (SYS) is intended for command files with filetypes of PRL or COM. Files with these types should be placed on the system drive (see Section 1) and assigned the special file attributes of System (SYS) arid Read Only (RO). The SYS attribute allows the command files to be accessed front any valid drive or user number even though they ere physically located in user 0 on the system drive. File attributes can be assigned using the SET utility described in Section 7.

If the [SYS] option is not used and system files do exist on the directory, DIR displays a "System Files Exist" message. In the example below, the A drive has been designated the system drive and contains a number of utility programs of type PRL.

```
OA>DIR
Directory for User 0:
A: DUMP      ASM : TESTI     ASM : TEST2     ASM : DEBLOCK ASM
A: DISKDEF   LIB
System Files Exist
```

With the (SYS) option, DIR also displays the system files.

```
OA>DIR [SYS]
Directory for User 0:
A: DIR       PRL : PIP       PRL : SUBMIT    PRL : ERAQ      PRL
A: ED        PRL : ASM       PRL : DDT       PRL : LOAD      PRL
A: STAT      PRL : TOD       PRL : DUMP      PRL : DUMP      ASM
A: TEST1     ASM : TEST2     ASM : DEBLOCK   ASM : DISKDEF   LIB
```

The system files are not normally displayed so they can be viewed as built-in to the MP/M II system.

**OA>DIR *.ASM [G8]**

Use the G option to get and display the directory from another user number. Follow G with the number of the user area you want to display. Valid user numbers range from 0 to 15. In the following example, the DIR command displays all the files of type ASM in user number 8 on the A drive.

```
OA>DIR *.ASM[G8]
Directory for User 8:
A: PROGRAM1 ASM : PROGRAM2 ASM : TEST       ASM
```

**OA>DIR *.ASM, *.LIB**

Multiple DIR commands can be given in one line. Each command must be separated from previous commands with a comma or space. In the above example, DIR first displays the ASM files and then the .LIB files from user 0 on Drive A. The SYS and G options affect all filenames in the command line.

## 6.2 The SDIR Utility

**Syntax:**

```
SDIR
SDIR d:
SDIR filespec
SDIR [option]
SDIR [option = modifier]
SDIR [option] d:
SDIR [option = modifier] d:
SDIR [option] filespec,filespec
SDIR [option = modifier] filespec,filespec
```

The following "exceptions" are allowed.

```
SDIR d: [option]
SDIR filespec [option]
SDIR filespec,filespec [option]
```

The SDIR utility is an enhanced combination of the DIR utility and the STAT utility. SDIR is equipped with all of the options needed to display MP/M II files in a variety of ways. SDIR can search for files on any or all drives, in any or all user areas.

SDIR supports only global options, those which modify the entire command line. Formal global options are allowed only after the command name on the command line. To be more "friendly," SDIR allows the option list to occur anywhere on the command line. However, only one option list is allowed.

Options must be enclosed in square brackets. The options can be used individually or strung together separated by commas. Only one or two letters arc needed to unambiguously identify the option. The right hand bracket is needed only if the option is followed by a drive or file specification. SDIR with no specified options displays files in the default user area on the default drive.

SDIR searches on single file specifications or on combinations of up to ten file specifications. As explained in Section 3, a file specification consists of an optional drive specification and colon, followed by a filename, followed by an optional period and filetype, followed by any necessary semicolon and password. The filename and filetype can include asterisks and/or question marks for wildcard searching. SDIR treats d: as if it were the ambiguous filespec, d:*.*.

The most efficient way to become familiar with SDIR is to use it. Type the command:

**OA>SDIR [HELP]**

to display a list of example SDIR commands. SDIR is a passive program. It doesn't change any of the information on the disk or anything vital in memory, so you can experiment with it freely.

## 6.3 SDIR Format

The SDIR utility has three formats. The first is the default or "full" format. The second format is the "size" format. The third format is the "short" format.

The default or "full" format shows the name of the file, the size of the file in number of records and in number of kilobytes, and the attributes of the file. If there is a directory label on the drive SDIR shows the password protection mode and the time stamps. If there is no directory label, SDIR displays two file entries on a line, omitting the password and time stamp columns. The display is alphabetically sorted.

The following is an example of an SDIR display. Since Drive M has no directory label, SDIR displays two files per line.

```
6E>sdir m:
00:20:24 E:SDIR       PRL     (User 0)

Directory For Drive M:    User  6

 Name       Bytes  Recs  Attributes Name      Bytes   Recs Attributes
----------- ------ ----- ---------- --------- ------ ----------------
FRONT  PRN     5k    34 Dir RW     FRONT TEX    3k       24 Dir RW
TOC    PRN     6k    46 Dir RW     TOC   TEX    4k       29 Dir RN

Total Bytes   =   18k  Total Records =     133   Files Found = 4
Total 1k Blocks = 18   Used/Max Dir Entries For  Drive M:4/ 64
```

SDIR displays the Read Only or Read Write, the Sys or Dir, the Archive and the user defined attributes (F1,F2,F3,F4) of a file. SDIR displays SYS if the System attribute of the f ile is on, and DIR if it is off. SDIR displays RO if the file is Read only and RW if it is not. SDIR displays the number 1, 2, 3, or 4 corresponding to the number of any user attributes that are on.

The full format displays two measures of file size. The size of the file in kilobytes is the total amount of disk space allocated to the file by the operating system. The number of records is the actual file length in 128 byte units. Depending on the size of a block on the disk, the operating system in general allocates more storage than is needed by the file. You can think of the disk block size as a minimum allocation unit.

The second format is the size format. SDIR [SIZE] displays the file name and the file size in kilobytes.

Both the full format and the size format follow their display with two lines of totals. The first line displays the total number of kilobytes, the total number of records, and the total number of files for that drive and user area. The second line displays the total number of 1k blocks needed to store the listed files. The number of 1k blocks shows the amount of storage needed to store the files on a single density diskette, or on any drive that has a block size of one kilobyte. The second line also shows the number of directory entries used per number of directory entries available on the drive. These totals are suppressed if only one file is found

The third format is the short format. The short format is similar to the DIR command except that SDIR [SHORT] also displays files with the System attribute on. The short format does not sort the files alphabetically. This is because the short format does not collect the files in memory. For this reason, the short format will never run out of memory and can display any size directory.

Table 6-1 explains each of the SDIR options.

**Table 6-1. SDIR Options**

| Command | Result |
|---|---|
| SDIR | displays all files on the default drive, in the default user area, in full format, sorted alphabetically. This command is the default display and is equivalent to SDIR [RO,RW,SYS,DIR,SORT,FULL,XFCB]. |
| SDIR [SYS] | displays only the files that have the SYS attribute on. |
| SDIR [RO] | displays only the files that have Read Only attribute on. |
| SDIR [DIR] | displays only the files that have the SYS attribute off. |
| SDIR [RW] | displays only the files that are set to Read Write. |
| SDIR [XFCB] | displays all the files that have Extended File Control Blocks (XFCBs). See the SET command in Section 7 for a discussion of XFCBs. The SHORT option is ignored when used with the XFCB option. |

**Table 6-1. (continued)**

| Command | Result |
|---|---|
| SDIR [NONXFCB] | displays those files without Extended File Control Blocks.The SHORT option is ignored when used with the NONXFCB option. |
| SDIR [USER=n] | displays files under the user number specified by n. |
| SDIR [USER=ALL] | displaysfiles under all the user numbers for the default drive SDIR [USER=(O,l .... 15)] |
| | displays files under the user numbers specified. |
| SDIR [DRIVE=d] | displays files on the drive specified by d. The drive specified must exist. DISK is also acceptable in place of DRIVE in all the DRIVE options. |
| SDIR [DRIVE=ALL] | Displays files on all of the "logged in" Drives. A "logged in" drive is a valid Existing drive that has been accessed since the last dskreset (see Section 5.1) |
| SDIR [DRIVE=(A,B,C .... P)] | displays files on the drives specified. The specified drives must exist. SDIR [FULL] |
| | displays the files in full format. if 110 directory label exists for the drive, or the NONXFCB option was specified, the columns for date and time stamps are omitted, and two files are displayed per line. This is the default display. |

**Table 6-1. (continued)**

| Command | Result |
|---|---|
| SDIR [LENGTH=n] | displays n lines of f iles before inserting a table heading. n must be in the range between 5 and 65536. |
| SDIR [SIZE] | d i splays the d i sk space in k i lobytes allotted to the files on the default or specified drive. |
| SDIR [FF] | The Form Feed option is used with the CONTROLP character to make hard copies of directories. It sends an initial form feed to the printer. If the LENGTH option is also specified, the printer issues a form feed every n lines. |
| SDIR [MESSAGE] | The, message option is used when SDIR is searching for files on more than one drive and/or user area. Normally, SDIR does not print the names of the drives and users it is searching. With this option SDIR displays the names of the specified drives and user areas and any files residing there. If there are no files in the specified locations, SDIR displays the "File not found" message. |
| SDIR [NOSORT] | SDIR normally sorts files alphabetically. SDIR [NOSORT] displays the files in the order it finds them on the disk. When the SHORT format is used NOSORT is automatically set. |
| SDIR [EXCLUDE] | SDIR with the EXCLUDE option displays the files on the default in the default user area that DO NOT MATCH the files specified in the command line. |

**Table 6-1. (continued)**

| Command | Result |
|---|---|
| SDIR [HELP] | displays examples of various SDIR commands. |
| SDIR [SHORT] | displays files in four columns and excludes password and time stamping columns. NOSORT is automatically implemented with this option and the files are not listed alphabetically. |

The examples below illustrate some of the uses of the SDIR command. The following command line instructs SDIR to list all the SYStem files of type PLI, COM and ASM on the system in the currently "logged" drives in any user area.

**OA>SDIR [user=all,drive=all,sys] *.PLI *.COM *.ASM**

The following example instructs SDIR to display the filename TESTFILE.BOB if it is found on any logged in drive or user area.

OA>SDIR [drive=all user=all] TESTFILE.BOB

The example below instructs SDIR to list each Read Write file that resides on Drive D with its size in kilobytes. Notice that d: is equivalent to d:*.*.

OA>SDIR [size,rwl D:

The following example lists all the PRL files on drive D that have XFCBs.

**OA>SDIR [xfcb] D:*.PRL**

The example below displays all the files on drives A, B and C in short format.

OA>SDIR [short] A: B: C:

The following SDIR command lists all the files on the default drive and user area that do not have a filetype of PRL or COM.

OA>SDIR [exclude] *.COM *.PRL

## 6.4 The STAT Command

Syntax:

```
STAT
STAT d:
STAT d: = RO
STAT filespec
STAT filespec [SIZE]
STAT filespec [RO]
STAT filespec [RW]
STAT filespec [SYS]
STAT filespec [DIRI STAT d:DSK:
STAT d:USR:
STAT VAL:
```

STAT can display or set the status of several different elements of your computer system. At your request, it reports the amount of free space left on a disk, the amount of disk space occupied by a file or group of files, and displays and sets file attributes.

The optional command tail tells STAT what to check or change. The command tail is the portion of the STAT command following STAT. If you just enter STAT, without a command tail, STAT displays the amount of free space on the default disk and any other disk accessed since the last DSKRESET:

```
OA>DSKRESET
OA>STAT
A: RW, Space:           74k
OA>TYPE B:LETTER.TEX
Compudealer
123 W. Fourth St.
Inglevale, CA
Dear Compudealer:

.
.
.

OA>STAT
A: RW, Space:           74k
B: RW, Space:        4,800k
```

In general, if you enter a drive specification in the STAT command, STAT displays the disk status or sets the disk status to RO. Section 6.4.1 describes STAT commands for disks. If you enter a file specification in the STAT command, STAT displays or sets file status.

### 6.4.1 Disk Attributes and Statistics

If you do not need a list of free space on all drives accessed since the last DSKRESET, but do need to know how much space is left on one disk, enter a drive specification for that disk in the STAT command as shown below:

```
OA>STAT B:

Bytes Remaining on B: 4,800k
```

Unlike the STAT command with no command tail, this does not display whether the drive is marked Read Only (RO) or is available for both reading and writing (RW).

STAT can change a RW drive to RO with the command shown below.

```
OA>STAT B:=RO
```

After setting the B drive to Read Only, the STAT command with no command tail reflects this change as shown below.

```
OA>STAT
A: RW, Space:        74k
B: RO, Space:     4,800k
```

After this sequence, drive B carries the RO attribute until you use the SET command to reset it to MN.

STAT can also give you a detailed description of how data is stored on a disk or disks. To request this report, enter the special argument DSK: with or without a drive specification in the STAT command as shown below.

```
OA>STAT DSK:
```

With a drive specification, STAT reports only on the drive indicated. Without a drive specification, STAT reports on all drives accessed since the last DSKRESET. The following example requests a report on a single drive:

```
OA>STAT B:DSK:

        B: Drive Characteristics
   65,536: 128 Byte Record Capacity
    8,192: Kilobyte Drive Capacity
      512: 32 Byte   Directory Entries
        0: Checked   Directory Entries
    1,024: Records   Directory Entry
      128: Records   Block
       68: Sectors   Track
        0: Reserved  Tracks
```

The left column of this report gives the values specific to the disk. The right column describes the data storage format. Most of this information is of interest ' only to a programmer modifying MP/M II's Basic Input Output System to change how data is stored on the disk. However, this display does tell you the maximum amount of data you can put on your disk (Kilobyte Drive Capacity) and the maximum number of files you can create on the disk's directory (32 Byte Directory Entries). Note that if your files are large, you might fill the disk with data before you fill the directory with file entries. Table 62 defines the elements of a STAT DSK: report.

**Table 6-2. STAT DSK: Disk Storage Format Report**

| Message | Meaning |
|---|---|
| 128 Byte Record Capacity | At maximum, you can store this number of 128 byte records on the disk. |
| Kilobyte Drive Capacity | At maximum, you can store this number of kilobytes on the disk. |
| 32 Byte Directory Entries | At maximum, you can create this number of files on the disk. |
| Checked Directory Entries | The number of directory entries checked each time the disk is accessed to verify that it has not been removed and replaced by another disk. For floppy disks, this is usually the same as 32 Byte Directory Entries. For non-removable hard disks, this is usually 0. |
| Records / Directory Entry | At maximum, a single directory entry can reference this number of records. |
| Records / Block | This is the minimum amount of space that must be assigned to a file on this disk. |

**Table 6-2. (continued)**

| Message | Meaning |
|---|---|
| Sectors / Track | Each track on the disk is divided into this number of 128 byte sectors. |
| Reserved Tracks | This number of disk tracks are not available for data storage. They are reserved for tile MP/M II system. |

### 6.4.2 File Attributes and Statistics

To display the attributes and statistics of a single file, enter the file reference in the STAT command as shown in the example below.

```
OA>STAT B:LETTER.TEX

Recs   Bytes FCBs Attributes        Name
  16    l6k,     1 Dir RW           B:LETTER    TEX

        l6k      1 (1 file, 2lk blocks)

       Bytes Remaining On B: 4,800k

OA>
```

This display reports that LETTER.TEX on drive B occupies 16 kilobytes of disk space grouped into 16 128 byte records. It uses one File Control Block (FCB) in the disk directory. FCB is the technical name for a directory entry. Large files may require several directory entries (FCBs) . The file has the DIRectory attribute, meaning that it is not a system file. it also carries tile RW access attribute, meaning that it is not write protected. No other special file attributes are set for the file.

To display the above, information for a group of files, enter an ambiguous file reference as the STAT command tail. To review the status of all files on tile disk, enter *.* as the filename. To select a group, enter the appropriate wildcard characters in the filename. For example, *.PRL in tile following command line selects .PRL files on the A disk:

**OA>STAT *.PRL**

```
 Rees   Bytes FCBs Attr ' ibutes      Name
   74     10k     1 Sys  RO         A:ASM.PRL
   50      7k     I Sys  RO         A:RDT.PRL
    4      1k     1 Sys  RO         A:DUMP.PRL
   68      9k     I Sys  RO         A:ED.PRL
   14      2k     1 Sys  RO         A:LOAD.PRL
   14      2k     I Sys  RO         A:DIR.PRL
   75     10k     1 Sys  RO         A:PIP.PRL
   78     10k     1 Sys  RO         A:STAT.PRL
   40      5k     1 Sys  RO         A:SUBMIT.PRL
   20      3k     1 Sys  RO         A:TOD.PRL
   29      4k     I Sys  RO         A:ERAQ.PRL
------------------------------------------------
          63k    1l (11 files, 63-1k blocks)

       Bytes Remaining On A: 74k

OA>
```

Under the dotted line, STAT displays the totals for the files listed. The total disk space consumed is under the bytes column. The total number of directory entries appears under the FCBs column. The number of files is listed in parentheses. The number of files might be less than the total FCBs if some files are large enough to have more than one FCB.

A second byte total is given within the parentheses. This total displays the number of 1k blocks used by the files listed in the STAT display. It represents the amount of space that would be consumed by the listed files if they were placed on a standard single density 8" floppy disk with the normal 1k block size. The 1k block total is useful when determining if enough space exists on a floppy disk to back up a group files from a hard disk. The number of blocks used on a hard disk for a given file might be greater than the number required for the same file on a floppy disk. This is because of the difference in the minimum amount of space allocated to any file (the block size).

Under MP/M and CP/M a file of even one character is allotted one whole block. One block is the minimum amount of space that can be allocated to a file, no matter how small the file is. This minimum can range from 1 kilobyte on a single density floppy to 16 kilobytes on some hard disks. Therefore, the number of blocks a file needs on a, floppy disk might be substantially less than the number needed on a hard disk. To help illustrate this, examine the following STAT display of the file LETTER.TEX again:

```
OA>STAT B:LETTER.TEX

 Recs   Bytes FCBs Attributes       Name
   16    16k    1 Dir RW          B:LETTER   TEX

         16k    1 (1 file, 2-1k blocks)

       Bytes Remaining On B: 4,800k

OA>
```

Note that although the file contains only 16 records it consumes 16k of disk space. Sixteen 128 byte records equals 2048 bytes or two kilobytes, yet the file takes up 16k of disk space because the block size on the B drive is 16k. The block size for a particular drive can be determined from the STAT DSK: display for that drive (see Section 6.3.1 above). The B drive in this example has 128 records (16k) per block.

In the example below, the file displayed has several special attributes.

```
OA>STAT B:TEST.TEX

 Recs   Bytes FCBs Attributes       Name
   38    16k    1 Dir RW XA1234 B:TEST      TEX
------------------------------------------------
         16k    1 (1 file, 5-1k blocks)

       Bytes Remaining On B: 4,800k

OA>
```

The Archive attribute (set by PIP when using the Archive Option [A]), and the user definable, attributes: F1, F2, F3 and F4 are all displayed in the STAT display under the Attributes column. if the file has been archived, STAT displays the letter A and if any of the user definable attributes are on, STAT displays the number of the attributes that are on. The user definable attributes can be set using the SET command. They are not used by MP/M II but can be used by application programs to mark particular files as desired. STAT also displays an X in the Attributes column if the displayed file has an XFCB (Extended File Control Block, necessary for password protected or time stamped files).

The STAT file display has a size option (Function 35 in the MP/M II Programmer's Guide) , that displays the computed size of each listed file in a Size colunin, as shown below.

**OA>STAT *.PRL [SIZE]**

```
    Size   Recs   Bytes FCBs Attributes       Name
      74     74     10k    1 Sys RO          A:ASM.PRL
      SO     50      7k    1 Sys RO          A:RDT.PRL
       4      4      1k    1 Sys RO          A:DUMP.PRL
      68     68      9k    I Sys RO          A:ED.PRL
      14     14      2k    1 Sys RO          A:LOAD.PRL
      14     i4      2k    1 Sys RO          A:DIR.PRL
      75     75     10k    1 Sys RO          A:PIP.PRL
      78     78     10k    1 Sys RO          A:STAT.PRL
      40     40      5k    1 Sys RO          A:SUBMIT.PRL
      20     20      3k    1 Sys RO          A:TOD.PRL
      29     29      4k    1 Sys RO          A:ERAQ.PRL
---------------------------------------------------------
                    63k   11 (11 files, 63-lk blocks)

                   Bytes Remaining On A: 74k

OA>
```

The computed size of a file is usually the same as the number of records listed for the file, unless the file is a random access file with some unwritten gaps in it.

MP/M II supports drives of up to 512 megabyte capacity. However, the STAT values in the Recs and Bytes columns are accurate only when the number of records and kilobytes do not exceed 65,535. For very large files exceeding 8 megabytes, you must use the SIZE option to display the correct size of the file. STAT can display a maximum of 512 files at. a time.

The example below, shows how STAT can assign the Read only attribute to a file. This means that the file is write protected (cannot be erased or changed) until the file is reassigned the normal Read Write (RW) attribute.

**OA>STAT *.PRL [RO]**

To further protect a file from user access, MP/M II supports another attribute, SYS. When a file is marked with the SYS attribute, a DIR command cannot list it at the console. The complement of the SYS attribute is the DIR attribute, (the default of all files created under MP/M II) To assign SYS or DIR status to a file, enter the attribute in square brackets after the file specification in the STAT command tail. You can use an ambiguous filename to assign an attribute to a group of files.

The following STAT VAL: display summarizes the possible STAT commands.

```
STAT VAL:
STAT 2.0

Read Only Disk: d:=RO
Set Attribute: d:filename.typ [RO] [RW] [SYS] [DIRI
Disk Status   : DSK: d:DSK:
User Status   : USR: d:USR:
```

The first line in the display shows how to assign RO status to the disk in the drive specified by d: . The Set Attribute line lists the f o u r attributes STAT can assign to the file specified by d:filename.typ. The third and fourth lines list the commands to enter for a report of disk or user status.

STAT can display the status of file storage under the 16 disk user numbers. When you enter the command STAT USR:, STAT reports your current user number and also which user numbers have files on them, as shown in the following display:

```
OA>STAT USR:

A: Active User     0
A: Active Files:   0  3  8
OA>
```

The option USR: can be prefaced with a drive to display the user number status of another disk.

In the MP/M II system, the functions of the STAT command have been separated into three new commands: SHOW, SET and SDIR. This division allows the separation of passive functions, such as displaying the status of files, from active functions, such as changing file or disk attributes. It also facilitates a more logical grouping of these functions. SHOW and SDIR are passive commands that can riot change anything on the disk or in memory. They merely display the characteristics of the files and drives in the MP/M II system. The, SET command, described in Section 7, controls the setting of file and disk attributes as well as password protection and time stamping.

## 6.5 The SHOW Utility

**Syntax:**

```
SHOW
SHOW SPACE
SHOW DRIVES
SHOW USERS
SHOW LABEL
SHOW HELP
SHOW d:
SHOW d:SPACE
SHOW d:DRIVE
SHOW d:USERS
SHOW d:LABEL
```

The SHOW utility provides the same information as the passive STAT functions, with the addition of the SHOW LABEL option. SHOW by itself displays the drive, the Read Only or Read Write mode for that drive, and the remaining space in kilobytes for all logged in drives in the system. The SHOW SPACE display is the same as the SHOW display. SHOW HELP displays a list of the SHOW options. SHOW with the optional drive specifier displays the SHOW information for the specified drive only, as shown in the following example.

```
OA>SHOW B:
B: RW, Space:      9,488k
```

The SHOW DRIVES command displays the drive characteristics of logged in drives on the system, or for a specified drive. The following is an example of the SHOW DRIVES display:

```
    A: Drive Characteristics
3,600:  128 Byte Record Capacity
  450:  Kilobyte Drive Capacity
   96:  32 Byte    Directory Entries
   96:  Checked    Directory Entries
  128:  Records    Directory Entry
   16:  Records    Block
   48:  Sectors    Track
    2:  Reserved   Tracks
```

The SHOW USERS command displays the current user number and all user areas on the drive that have files assigned to them, as shown in the example below:

```
OA>SHOW USERS
Active User : I
Active Files: J 2 3 4
```

The SHOW LABEL command returns a display of the optional directory label, if it has been created. The directory label is an entry within the directory and contains information that describes special attributes of the disk to the operating system. For example, the label tells whether time stamping and password protection are turned on for that disk.

You can give the label a. name to help identify the data that is stored on that disk. You can assign a password to the label to prevent unauthorized access to the SET label functions that turn on or off time stamping and password protection.

You create a directory label when you use a SET command to turn on password protection or time stamping for a drive. Section 7 explains the SET commands in detail. The following example illustrates the SHOW LABEL display:

```
Label for drive B:

Directory Passwds Make   Stamp   Stamp  Label Created  Label Updated
Label      Reqd   XFCBs  Create Update
--------- ------- ------ ------- ------ -------------  --------------
TOMSD1SK.   On     on     on      on    07/04/81 10:30 07/08/81 09:30
```

The first column, Directory Label, displays the name assigned to that drive directory by a SET [NAME=TOMSDISK] command. The second column, Passwds Reqd, shows that password protection has been turned on for that drive with a SET [PROTECT=ON] command. if password protection is turned off, all password protection on the drive is temporarily deactivated. This means that you can access, update and even delete any password protected files on the drive even though you do riot know the password. obviously, this option is normally set on. It should only be set off if the password to an important file is forgotten. If password protection is used, be sure to assign a password to the label itself so that an unauthorized person cannot turn off password protection.

The third column, Make XFCBs, shows that the automatic creation of Extended File Control Blocks (XFCBs) has been turned on for the drive with a SET [MAKE=ON] command or one of the SET time stamping commands. For a file to have space for a password and two time stamps, the file must have an Extended File Control Block and the directory must have a label. When the Make XFCBs option is on, all files created or copied to the drive are automatically given an XFCB. Otherwise, a specific group of files can be assigned XFCBs with the SET command.

As mentioned above, each file can have up to two time stamps. The first of these time stamps can be either the creation date and time for the file or the time and date of the last access to the file (access is defined as reading from or writing to the file) . The fourth column cf the SHOW LABEL command displays both the type of stamp and whether or not it is on. In the example above, creation time stamps are given to new files as shown by the Stamp Create column heading. Creation time stamps can be turned on for the drive with a SET [CREATE=ON] command.

If access time stamps are preferred, they can be turned on for the drive with a SET (ACCESS=ON] command. In this case, the fourth column heading in the SHOW LABEL display is Stamp Access instead of Stamp Create. It is best to decide whether you want creation date stamping for files or access date stamping for files, set the appropriate mode, and leave it that way. Otherwise, once changed from ACCESS back to CREATE, ACCESS dates for files may be in the Stamp Create field.

The fifth column displays the status of the second time stamp field, the update time stamp. Update time stamps display the time and date of the last update to a file, that is, the last time someone wrote to the file. In the display above, update time stamps have been turned on with a SET [UPDATE=ON] command.

In addition to showing the password protection, make XFCBs mode, and the active time stamps on a drive, SHOW LABEL also displays the date and time that the label was created and last updated.

# Section 7
# The SET Command

## 7.1 Introduction to the SET Command

Syntax:

```
SET d:[RW]
SET d:[RO]
SET [option=modifier]
SET d:[option=modifier]
SET filespec [option=modifier]
SET filespec [option]
```

The SET command initiates password protection and time stamping of files in the MP/M II system. It also sets file and drive attributes, such as the Read Only, System, and user definable attributes. This section discusses three major topics: password protection, time stamping, and setting file attributes.

The SET command always requires a parameter consisting of an object and an action. The object can be either a drive, a file or a group of files. The action can be any of a number of options described in this section. The options are always enclosed in square brackets. If no drive or filename is specified, the default drive is assumed.

The SET command includes options that affect entire drives and options that effect files or groups of files. SET options can be strung together and separated by commas within the square brackets. Note that spaces before or after the brackets and equal sign are optional. Multiple SET file and drive commands can be specified in one command line if they are separated by commas.

## 7.2 Password Protection

The MP/M II Operating System supports password protection of files. Passwords can be up to 8 characters long and are required for access to the file to which they are assigned. Passwords can be assigned to the MP/M II commands present on your system (PRL and COM files). When accessing a password protected file or command, the password must be preceded by a semicolon and typed as part of the command or file name as shown below.

**OA>TYPE B:DOCUMENT.LAW;SECRET**
**OA>ERA;SECRET B:DOCUMENT.LAW;SECRET**

In the first example_ above, the file DOCUMENT.LAW is protected by the password SECRET. The password must be included in the file specification to type the file. in the second example, both the

MP/M II ERA command (See Sect ion 9. 2) and the file DOCUMENT. LAW are protected by the password SECRET. Some MP/M II commands prompt for the password if it is missing from the file specification. For instance, in the ERA command above, the password for B:DOCUMENT.LAW could have been left off as shown below.

**OA>ERA B:DOCUMENT.LAW**

Not erased: B:DOCUMENT.BAK Password Error
Password ? (enter SECRET here, it is not echoed)

OA>

To assign passwords to files on a drive, you must first turn on password protection for the drive. You can determine if password protection has been turned on with the SHOW LABEL command discussed in Section 6.5.

### 7.2.1 Turning On Password Protection

SET controls password protection for individual files and for the directory or drive label. The optional directory or drive label is an entry within the directory. Just as a directory entry describes a file to MP/M II, the label contains information that describes special attributes of the disk to the operating system. For example, the label tells MP/M II whether or not time stamping and password protection are turned on for that disk. You can give the label a name to help identify the data that is stored on a given disk.

You can assign a password to the label. If the label has no password, any user who has access to the SET program can set the drives to Read Only or turn on time stamping or password protection for the whole drive with a single command. If you assign a password to the label, then you must supply the password to set any of the functions controlled by the label. SET always prompts for the password.

It is extremely important that you assign a password to the directory label if you intend to use the password protection features of MP/M II. Because of its power, it is useful to think of the directory label password as a kind of super password. if you know the super password, you can turn password protection off for the whole drive and thereby gain access to all files on the drive, even those which are protected by passwords you don't know.

**OA>SET [PASSWORD=SECRET]**
**OA>SET [PASS=<cr>**

The above commands assign passwords to the directory or drive label. In the first command, the password "SECRET" is assigned. This limits access to the SET label functions. The second command nulls the existing password. You simply type a carriage return for the password. The drive options themselves are very powerful and should usually be protected with a password. The only way to turn a label password off is to set the password to carriage return. The following example illustrates setting the label password to "secret", and then turning password protection on for the drive.

```
OA>SET [PASSWORD=SECRET]

Label for drive A:

Directory        Passwds   Make      Stamp     Stamp     Stamp
Label            Reqd      XFCBs     Create    Access    Update

A:Label            off       off       off       off       off

Password = SECRET

OA>SET [PROTECT = ON]

Password ? (Enter SECRET here, it is not echoed) Label for drive A:

Directory        Passwds   Make      Stamp     Stamp     Stamp
Label            Reqd      XFCBs     Create    Access    Update

A:Label            on        off       off       off       off

OA>
```

When setting drive options, SET always displays the settings of all the drive options in a status display similar to that of the SHOW LABEL command described in Section 6.5. After a directory label has been created by setting a drive option or setting the directory label password, SET always prompts for the label password when you change any of the label options. In the above example, the password prompt occurs in the second SET command, when password protection is turned on.

The password protection option must be turned on before assigning passwords to individual files or commands. The only exception to this is the directory label password itself. Since it is the "super password" controlling password protection, it is always required. If no password has been assigned to the directory label, or if the password is just a carriage return, then simply press the carriage return key after the "Password ?" prompt. In the above example, because the password is "secret", you type the word "secret" after the prompt. Note that this password is not echoed on the console. if you make a mistake, you can use a ^X to erase the line and start over, or a ^H to erase the previous character. (See Section 2. 1 for ~,n explanation of these line editing control characters).

The password protection option can be turned off at any time with a SET [PROTECT = OFF] command. Once a label password has been assigned and the PROTECT option has been turned ON, you are ready to begin assigning passwords to files.

### 7.2.2 Assigning Passwords to Files

**OA>SET MYFILE.TXT [PASSWORD = mypsword]**

The PASSWORD option of the SET command sets the password for the specified file to the password indicated by "your password". Passwords can be up to eight characters long. SET treats upper and lower case passwords identically.

Note: It is strongly advised that you write down or remember the passwords that you have assigned to your files. If you do not remember the passwords that you have assigned to your files, you will not be able to access those files unless password protection is turned off for the whole drive. if you assign a password to the directory label and then forget the password, you won't be able to turn off password protection for the drive.

```
OA>SET MYFILE.TXT [PROTECT = READ]
OA>SET MYFILE.TXT [PROTECT = WRITE]
OA>SET MYFILE.TXT [PROTECT = DELETE]
OA>SET MYFILE.TXT [PROTECT = NONE]
```

When a password is assigned to a file, there are three possible levels of protection afforded to the file by its password (see Table 71). The protection modes are READ, WRITE, DELETE and NONE (no protection). When the PROTECT mode is set to READ, you need the password even to read from the file. When the PROTECT mode is set to WRITE, you do not need a password to read from the file, but you do need a password to write to or make any changes to the file. If the PROTECT mode is set to DELETE, then you do not need the password to read from or make changes to the file, but you do need a password to erase or rename the file. Finally, when the PROTECT mode is set to NONE, SET erases the password. Files without a password or with a password of blanks always have a protection mode of NONE. Files assigned a password for the first time default to the protection mode of READ.

**Table 7-1. MP/M II Password Protection Modes**

| Mode | Protection |
|---|---|
| READ | The password isrequired for reading, copying, writing, deleting or renaming the File. |
| WRITE | The password is required for writing, deleting or renaming the file. |
| DELETE | The password is only required for deleting or renaming the file. |
| NONE | No password exists for the file. |

The SET PROTECT and SET PASSWORD commands can refer to ambiguous filenames. In the example below, the password SECRET is assigned to all the TEX files on drive B . Each TEX file is given a protection mode of WRITE to prevent unauthorized editing.

```
OB>SET *.TEX [PASSWORD = SECRET, PROTECT = WRITE]

B:ONE        TEX  Protection = WRITE, Password = SECRET
B:TWO        TEX  Protection = WRITE, Password = SECRET
B:THREE      TEX  Protection = WRITE, Password = SECRET

OB>
```

SET only displays th password when a new password is assigned. If you attempt to changethe protection mode of a password protected file and do not include the password in the file reference, SET prompts for the password. An advantage to this is that when MP/M II prompts for the password, it is not echoed to the console and so can not be read by other people watching the screen.

### 7.2.3 The Default Password

MP/M II supports an additional password facility called the "default" password. You can set a default password that will be automatically tried whenever a password is required for a file or directory label operation. The default password only applies to the console at which it is set.

The default password is a convenience when your files all have the same password. When you begin working, you simply set the default password to your password, and from then on, you can access and edit your files without having to enter the password with each file specification. The files are still protected from access by any other users on the system.

```
OA>SET [DEFAULT = password]
```

The above command assigns a default password for your console. After setting the default password, when a password protected file is accessed, the system first tests any password delimited by a semicolon immediately after the filename. If no password is given or if the password is incorrect, the system tries the default password. If the default password is the same as the password assigned to the specified file, then that file can be accessed.

## 7.3 Time Stamping of Files

MP/M II can keep a record of the time and date of the last access and update of a file. Alternatively, the time of creation and last update may be recorded. To enable time stamping, you must turn on the time stamping option for the drive containing the files you wish to time stamp. If the files to be stamped already exist on the drive, you must also individually turn time stamping on for each file.

MP/M II supports three kinds of date/time stamps.

- CREATE date/time stamp
- ACCESS date/time stamp
- UPDATE date/time stamp

Although there are three kinds of date/time stamps, there can be at most two date/time stamps associated with a given file at one time. The two date/time stamps are recorded in two special fields which can be set up for a file. once a file has been assigned these fields, the time stamps can be displayed using the SDIR command (see Section 6.2).

You can choose to have either a CREATE date or an ACCESS date for files on a particular drive. You should decide whether you want to see the date/time of creation or date/time of last access to your files. Note, do not change this choice unless you reformat the disk or erase all files on the disk. otherwise you may forget whether the date on a particular file indicates the date the file was created or the last date that a user simply accessed the file.

Time stamping must be separately turned on for each drive desired. When you turn time stamping on, it is initiated for all new files created or copied to the drive. Alternatively, you can set up access and update time stamping for a particular group of files on the drive.

### 7.3.1 Time Stamping of New Files

When time stamping is turned on for a drive it automatically applies to all new files subsequently created or copied to the drive. Section 7.3.2 describes how to begin time stamping files which are already on the drive.

The time stamping option is controlled by the directory label. If the directory label is password protected, as recommended in Section 7.2.1, then you need to know the directory label password in order to initiate or change the time stamping options. Each of the three time stamping options can be turned on separately. Remember, though, that ACCESS and CREATE stamps cannot both be on at the same time.

**OA>SET [CREATE = ON]**

The above command turns on CREATE time stamps on the default drive. The CREATE date/time stamp records the time of file creation on the drive. A file can be created on a drive using an editor to create a new file, or by using the PIP command (Section 11) to copy the file to the drive. To record the creation time, the CREATE option must be turned on before the file is created. In the example shown below, three TEX files were copied to the B drive after using the SET [CREATE = ON] command to establish creation date/time stamping. The SDIR command is used display the file creation times.

**)B>SDIR**

```
Directory For Drive B:

    Name       Bytes   Recs  Attributes   Prot   Update  Create
------------------------------------------------------------------------
ONE      TEX      9k     71  Dir RW       None           08/03/81 10:56
THREE    TEX     12k     95  Dir RW       None           08/03/81 10:57
TWO      TEX     10k     76  Dir RW       None           08/03/81 10:56
```

The time stamps consist of date, hours and minutes using a 24 hour clock.

**OA>SET [ACCESS = ON]**

The above example shows how to turn on access time stamps if they are desired instead of creation time stamps. When the SET ACCESS option is turned on, the CREATE option is automatically turned off. The field heading changes from CREATE to ACCESS in the SDIR display, and any file accessed (read from or written to) after that has the date of access entered in the access field.

The example below shows an SDIR display of the three TEX files after access time stamping was turned on for a week.

```
OB>SDIR

Directory For Drive B:

    Name       Bytes   Recs   Attributes   Prot Update        Access
--------------------------------------------------------------------
ONE      TEX      9k     71   Dir RW       None            08/03/81 10:56
THREE    TEX     12k     95   Dir RW       None            08/05/81 15:45
TWO      TEX     10k     76   Dir RW       None            08/10/81 09:13
```

The access time stamps displayed show the time the file was last displayed or edited. Note that merely displaying a filename in a directory listing does riot constitute an access and is not recorded.

**OA>SET [UPDATE = ON]**

The above command turns on update time stamps. Update time stamps record the time the file was last modified.

If you are displaying both update and create date/time stamps, you will notice that editing a file changes both the update and create time stamps. This is because the MP/M II editor (see Section 12) does not update the original file but instead renames it with the file type BAK and creates a new version with the same name as the original.

The files shown below are updated by an accounting system. The accounting system does not recreate the files but simply writes additional information to them. This example shows a display of these files after turning on update time stamping.

```
OB>SDIR

Directory For Drive B:

     Name    Bytes  Recs  Attributes Prot Update          Create
----------- ------ ----- ---------- ----- -------------- ------------
GENLED  .DAT 109k  873   Dir RW     None  08/05/81 14:01 08/01/81 09:36
RECEIPTS.DAT  59k  475   Dir RW     None   08/08/81 12:11 08/01/81 09:40
INVOICES.DAT  76k  608   Dir RW     None   08/08/81 08:46 08/01/81 10:15
```

### 7.3.2 Time Stamping of Existing Files

To initiate time stamping of an existing file or group of files on a drive, two steps are required. First, the time stamping options for the drive must be turned on as described in the previous section. Second, time stamp fields must be provided for the files to be time stamped.

**OA>SET MYFILE.TXT [TIME]**

The above command provides date/time stamp fields for a particular file or files. As discussed in the previous section time stamps are placed in two special time stamp fields which must exist for each file that is time stamped. These fields are located in a special directory entry called the Extended File Control Block (XFCB). XFCB

The three time stamping options:

OA>SET **[CREATE = ON]**
**OA>SET [ACCESS = ON]**
**OA>SET [UPDATE = ON]**

each turn on a special drive option that automatically provides time stamp fields for all new files created or copied to the drive. This option is called the MAKE XFCB option. The MAKE XFCB option automatically makes XFCBs every time a file is created. It can be independently controlled as shown in the example below.

OA>SET [MAKE = OFF]

If only a particular group of existing files need time stamps, you may wish to turn the IMAKE XFCB option off to prevent the unnecessary creation of extended. file control blocks (XFCBs) for other files. The MAKE XFCB option must be ON if CREATE date/time stamps are desired because it assures that a time stamp field will be available at the time a new file is created.

## 7.4 Setting File and Disk Attributes

In MP/M I I , Jrives and files have a number of special attributes that can be set with the SET command. The attributes fall into two categories: access attributes and identification attributes. identification attributes

The access attributes determine if files and drives can be read and written (Read Write), or can only be read from and not written to (Read Only). Another access attribute determines if they are system files or not (system files in user number 0 can be accessed by all users).

Identification attributes are special attributes which can be used to help organize your disks and files. These include a special attribute called the Archive attribute that indicates that a file has been backed up (archived).

### 7.4.1 The Read Only Attribute

The Read only attribute has two modes: RO (Read Only) and RW (Read Write).

**OA>SET B:[RO]**
**OA>SET B:[RW]**

The above SET commands set the specified drive to Read Only or Read Write. If a drive is set to Read Only, PIP cannot copy a file to it, ERA cannot delete a file from it, REN cannot rename a file on it. You cannot perform any operation that requires writing to the disk. If a drive is set to Read Only and a process tries to write some data to a file on that drive, the system returns an error message and the process is aborted. When the specified drive is set to Read Write, you car, read or write to the disk in that drive.

**OA>SET MYFILE.TXT [RO]**
**OA>SET MYFILE.TXT [RW]**

The SET commands s,own above set the file specified by filespec to Read Only or Read. Write. When a file is Read Only, that particular file can only be read from and not written to. When a file is set to Read Write, the file can be read from or written to.

### 7.4.2 The System Attribute

The System attribute applies only to files. It has two modes: SYS (System) and DIR (Directory).

**OA>SET MYFILE.TXT [SYS]**
**OA>SET MYFILE.TXT [DIRI**

The SYS option of thc SET command sets the file specified by filespec to the type of SYStem. The DIR option sets the file to the type DIRectory. There are certain drives and user areas from which files can be accessed. When a command file has the attribute of SYS End is in user 0 of the system drive, it can be accessed from any location in the system. When a file has an attribute of DIR, it can be accessed only if it 14,' in the default user area and default drive (shown in the MP/M system prompt) . Additionally, system files are not normally displayed 5y the DIP command (See Section 6.1).

### 7.4.3 The Archive Attribute

The Archive attribute applies only to files. When the Archive attribute of a file is on it means that the file has been backed up (archived). There are two ways to turn the Archive attribute on. The Archive attribute is turned on by PIP when copying a group of files with the PIP [A] option (see Section 11). This PIP option requires an ambiguous file specification and copies only files matching the filespec that have not been copied, that is, files that have been edited or changed since the last time they were backed up with the PIP [A] option. PIP then sets the archive attribute on for each file successfully copied. The archive attribute is displayed by both STAT and SDIR (see Section 6).

The Archive attribute of a file. or files can be explicitly set or reset with the SET commands similar to those shown below.

```
OA>SET MYFILE.TXT [ARCHIVE = ON)
OA>SET MYFILE.TXT [ARCHIVE = OFF]
```

### 7.4.4 The User-Definable Attributes

Four user definable file attributes Fl, F2, F3 and F4, are supported to allow additional identification or classification of certain files. These attributes are not used at all by MP/M II and exist solely for one use of special applications or your own purposes. They are displayed by STAT and SDIR (see Section 6) and can be set with the following SET commands:

```
OA>SET MYFILE.TXT [Fl = ON]
OA>SET MYFILE.TXT [Fl = OFF]
```

In the example above , you can substitute F2, F3 or F4 for Fl to set or reset the file attributes Fl, F2, F3 and F4.

### 7.4.5 Naming Disks

When dealing with many floppy disks, it is often useful to name or number each disk. MP/M II provides a facility for this by creating a directory label for each disk. The directory label can be assigned an eight character name and three character type similar to E filename and filetype.

```
OA>SET [NAME=lablname.typ]
```

The NAME option assigns a name to the drive label. Label names make it easier to catalogue disks and keep track of different disk directories. If one of the other SET label options is entered before the SET NAME option, the label is created with the default name: "Label".

## 7.5 The SET Help Option

Similar to SHOW cind SDIR, SET has a help option which displays a number of examples of valid SET commands. Invoke the HELP display with the command:

**OA>SET [HELP]**

## 7.6 Additional Examples

Some examples of possible SET commands follow.

OA>SET *.PRL [SYS,RO,PASS=l23,PROT=READ] OA>

The above setting affords the most protection and the most difficulty for users. Because the password protection mode has been set to READ, you cannot even read one of the PRL files without entering the password 23, unless the default password has been set to 123. Even if the correct password is entered, you still cannot write to the file because the file's READ ONLY (RO) attribute is set ON. Finally, these files will riot be displayed by DIR unless the [SYS] option is used (see Section 6.1).

After the above SET command is executed, the following command reverses the protection and access attributes.

OA>SET ***.PRL [RW,PROT=NONE,DIR]**

```
Password ?123 <cr>
OA>
```

After executing the abc,,,e command, there is no password protection, the files of type PRL can be read from or written to, and will be listed in all DIR disp',iys. However, they can not be accessed from other user numbers or `,rives.

The example below sets all the files of types PRL and COM to Read Only and SYS. This is the recommended setting for all command files on your system. When set to Read Only and System, the files need only be placed in user 0 of the system drive and they will be available to all users on the system. The Read Only attribute protects the commands from accidental erasure.

```
0A>SET *.PRL [RO,SYS], *.COM [RO,SYS]

A:ABORT PRL set to system (SYS), read only (RO)
A:ASM PRL set to system (SYS), read only (RO)
A:CONSOLE PRL set to system (SYS), read only (RO)
A:DIR PRL set to system (SYS), read only (RO)

A:USER      PRL  set to system  (SYS), read only  (RO)
A:DDT       COM  set to system  (SYS), read only  (RO)
A:GENHEX    COM  set to system  (SYS), read only  (RO)
A:GENMOD    COM  set to system  (SYS), read only  (RO)
A:GENSYS    COM  set to system  (SYS), read only  (RO)
A:LOAD      COM  set to system  (SYS), read only  (RO)
A:MPMLDR    COM  seL to system  (SYS), read only  (RO)
```

# Section 8
# MPMSTAT, ATTACH, ABORT

## 8.1 The MPMSTAT Command

**Syntax:**

MPMSTAT

The MPMSTAT command displays a long list of the status of various system factors, such as the number of consoles. A complete discussion of the listing of MPMSTAT is beyond the scope of this manual. However, the display does show the following information, some of which is useful to the casual user.

- Status information, including the number of consoles
- Processes that are ready to execute
  Processes waiting to read from an empty queue Processes waiting to write to a full queue
  Processes waiting on a time delay
  Processes waiting for a polled I/0 event Processes waiting for an interrupt
  A list of all the queues in the system
  Console number and the process attached to it Console number and processes waiting for it
  Printer number and the process attached to it Printer number and the process waiting for it Names of the programs allocated to each memory segment Console number from which the process originated, in brackets

The following example shows a typical MPMSTAT display. Note the Memory Allocation display at the end of the MPMSTAT example. The Memory Allocation display shows the base address of the particular memory segment, the size of the given segment, the number of the bank, the program allocated to that segment, and, in square brackets, the number of the console from which the process was initiated. Capital letters indicate names of programs or queues that are accessible from the console.

Note: The MPMSTAT display is more than one full screen on most consoles. To freeze the display long enough to read it you must enter a ^S. After reading, enter a ^Q to restart the display. Once a TS has been entered, only a ^Q or a ^C has any effect; other characters simply beep.

```
Top of memory = FFFFH
Number of consoles = 03
Debugger breakpoint restart # = 07
Stack is swapped on BDOS calls
memory is bank switched
BDOS disk file management is bank switched
Z80 complementary registers managed by dispatcher

Ready Process(es):
  MP/MSTAT [11 Idle
Process (es)DQing:
  [CliQ        ] CLI
  [ATTACH      ATTACH
Process(es) NQing:
Delayed Process(es):
Polling Process(es):
  TmpO
Process(es) Flag Waiting:
  01  Tick
  02  Clock
Flag(s) Set:
  03
Queue(s):
  CliQ      ATTACH    MXList    MXDisk
Process(es) Attached to Consoles:
  [01  TmpO
  [11  MPMSTAT
  [2]  Tmp2
Process(es) Waiting for Consoles:
  [1]  Tmpl        [1] ERAQ      [1] TYPE
Process(es) Attached to Printers:
  [0]  MPMSTAT
  [1]  Unattached
Process(es) Waiting for Printers:
```

Memory Allocation :

| | | | | |
|---|---|---|---|---|
| Base = 8COOH Size = 7400H Bank = OOH Allocated to MP/M II | | | | [0] |
| Base = OOOOH | Size = COOOH | Bank = 01H | Allocated to MPMSTAT | [1] |
| Base = OOOOH | Size = COOOH | Bank = 02H | Allocated to TYPE | [1] |
| Base = OOOOH | Size = 8COOH | Bank = OOH | Allocated to ERAQ | [1] |

This particular table shows memory bank 0, bank 1 and bank 2. Most of bank 0 is allocated for the MP/M II operating system. Some of Bank 0 remains and can be allocated to another process, as it is here. In this case, Bank 0 contains MP/M II and the ERAQ program. Bank I is allocated to MPMSTAT. The following example shows the Memory Allocation display again after entering several TD characters to reattach TYPE and ERAQ to console 1. This time MPMSTAT is allocated to bank 0. Banks I and 2 are free.

Memory Allocation:

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Base | = | 8COOH | Size | = | 7400H | Bank | = | OOH | Allocated to MP/M-80 | [0] |
| Base | = | OOOOH | Size | = | COOOH | Bank | = | 01H | * Free * | |
| Base | = | OOOOH | Size | = | COOOH | Bank | = | 02H | * Free * | |
| Base | = | OOOOH | Size | = | 8COOH | Bank | = | OOH | Allocated to MPMSTAT | [1] |

You can experiment with MPMSTAT to learn more about how MP/M II operates. Invoke a program that requires console 1/0, such as DIR for a directory listing. Detach the program from the console using the TD. Run MPMSTAT to see where your program is stored. Reattach the program to the console using TD again. Run MPMSTAT again.

## 8.2 The ATTACH Command

**Syntax:**

    ATTACH programname

MP/M II supports multi-programming at each console. This means that one program after another can be initiated at a given console without waiting for the previous program to actually finish running. However, only one program at a time can use the console for input or output. Depending on the memory size, any number of programs may be currently executing without affecting the console. Note that each program requires a memory segment, so if no memory segment is free then the program (process) cannot run.

Any process started at a console is automatically attached to that console and remains so until the command to detach the process is entered. The control character TD instructs a process to detach from the console. if the process does not require the console for input or output, it finishes executing. Execution of an MPMSTAT command shows that the program is running, which memory segment it is using, and from which console it was initiated. if the detached program does need the console to continue executing, processing must stop and the detached program must wait in a line or queue until a command is issued to reattach it to the console.

The ATTACH command attaches a process to the console at which the command is entered. Processes must be reattached to the same consoles from which they were detached. The ATTACH command attaches the program designated by programname to the console. It does not matter if that program is officially next in line for the console.

TD attaches as well as detaches programs from the console at which the ^D is entered. When ^D reattaches a program to the console, it does so on a first detached, first reattached basis.

An attempt to attach a program which has not been previously detached results in the following message.

    Attach failed

An attempt to access a file that has been opened by another process results in an open file error message. The following example illustrates a possible exchange using the ATTACH command.

```
3C>DUMP E:TYPE.PRL
0000 00 90 01 00 00 00 0C 00 00 00 00 00 00 00 00 00
00A 00 00 D
3C>E:TYPE DOCUMENT.TEX
Bdos Err on E: File Currently Open
Bdos Function: 15 File: TYPE.PRL
3C>ATTACH DUMP
00 00 00 00 0C 00 00 ...
02EO 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
02FO aO 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Attach: TmpO

<cr>
3C>TYPE DOCUMENT.TEX Dear Sir:

The company is pleased to announ ^D
Attach: TmpO
```

After ^D is struck twice, the system returns the message

```
Attach: Tmpn
```

where n is the number of the console to which the program was attached. This is essentially another system prompt. At this point, you can enter any valid MP/M II command, or press the carriage return key to display the MP/M III system prompt. (Note: The special line erase character, TX, might not work properly until you enter a carriage return.)

## 8.3 The ABORT Command

Syntax:

```
ABORT programname
ABORT programname n
```

The ABORT command immediately stops execution of the program specified by programname. The command can be entered at any console. To use an ABORT command to abort a program at the same console from which it was initiated, the program must first be detached from the console. To abort a program from any other console, the number of the console to which the program is attached must be substituted for the optional n. If ABORT cannot be executed the following message is displayed.

```
Abort failed
```

The following example illustrates a possible exchange using the ABORT command. The ABORT command is aborting the process TYPE running on Console number 1. The ABORT command is executing from another console. The user number does not affect ABORT.

```
0A> TYPE DOCUMENT.TEX

Dear Sir:

The company is pleased to inform you tha

5B>ABORT TYPE I <cr>
5B>
```

In the above example, assume that the TYPE command was issued from console number 1. The TYPE command is aborted from console number 3. To determine which console number is associated with the program you want to abort, you cEn use the MPMSTAT command described in Section 8.1.

The ^C character also aborts a running process, but only if it is still attached t,,) the console. When you press ^C, the system temporarily halts the program and responds with the message:

```
ABORT (Y/N)?
```

If you enter a Y, the program is immediately aborted. If you enter an N, the program continues executing. Generally, use the ABORT command with programs that have been detached from the console. Use ^C with programs that are currently attached to the console.

Note that you cannot abort Resident System Processes.

# Section 9
# TYPE, ERA, ERAQ, REN

## 9.1 The TYPE Command

**Syntax:**

```
TYPE filespec
TYPE filespec [PAGE]
TYPE filespec [Pn]
```

TYPE displays the contents of an ASCII file at the console. if the file contains tab or ^I characters, TYPE expands them using tab stops set at every eighth column. You must enter a specific filename as a command tail . An ambiguous filename or no filename at all results in an error message. Enter a drive specification if the file you want to examine is not on the default disk. Change, the user number in your system prompt with the USER command described in Section 5.2 of this manual if the file you want to examine is in a different user area. Enter the password if the file you want to type is password protected. If the password is omitted from the command line, TYPE prompts you for it. The password is not echoed at the console where you type it in. it is invisible.

If the TYPE command cannot find the file you want to type, it returns a "No file" error message.

If the contents of your file is longer than your console can display at one time , you might need to enter a ^S to temporarily halt the console listing before the top scrolls past you. Enter a ^Q o restart the listing.

TYPE can also print out the contents of your file on paper. Enter ^ before
pressing the carriage return key, and the file lists at the printer as well as at the console. This slows the console display to the speed of the printer.

Use TYPE to examine text. files, program source code, or other ASCII files formatted for human understanding. If you TYPE a file formatted for the computer's understanding , a COM or . PRL file for example, unintelligible characters appear on the screen.

TYPE with the (PAGE] option causes the console listing to automatically stop after listing twenty four lines of text. Press any character to restart the listing for the next twenty four lines of text.

You can use the [Pn] option to specify a page length other than twenty four. Substitute the number of lines per page for n in the command .

## 9.2 The ERA Command

Syntax:

```
ERA filespec
ERA filespec[XFCB]
```

ERA deletes a file specification from the directory, thus freeing the disk space occupied by the file. ERA accepts either a specific filename to delete one file, or an ambiguous filename to delete a group of files. Use the ERA command carefully, especially when erasing multiple files. It is a good idea to make a backup copy of your disk before erasing unwanted files because a simple typing error in an ERA command tail can have disastrous results!

The following example shows how an ERA command eliminates the filename NEWPROG.BAK from the current disk directory.

```
OB>DIR
Directory for User 0
B:  DOCFILEl TXT     DOCFILE2 TXT    : DOCFILE3 TXT    : NEWPROG ASM
B:  NEWPROG  BAK     ADDPROG ASM     : DOCFILEI BAK    : CHECKPRG ASM
B:  DOCFILE3 BAK     CHECKPRG BAK    : ADDPROG BAK     : DOCFILE2 BAK

OB>ERA NEWPROG.BAK

OB>DIR
Directory for User 0
B: DOCFILEl TXT : DOCFILE2 TXT : DOCFILE3 TXT : NEWPROG ASM
B: ADDPROG  ASM : DOCFILEI BAK : CHECKPRG ASM : DOCFILE3 BAK
B: CHECKPRG BAK : ADDPROG BAK  : DOCFILE2 BAK
```

Add a disk specification if the file you want to erase is not on your current disk. For example, the following sequence deletes NEWPROG.BAK when A is he current drive:

```
OA>ERA B:NEWPROG.BAK
OA>DIR B:
Directory for User     0
B: DOCFILEI rXT : DOCFILE2 TXT  : DOCFILE3 TXT : NEWPROG ASM
B: ADDPROG ASM :  DOCFILE1 BAK  : CHECKPRG ASM : DOCE'ILE3 BAK
B: CHECKPRG BAK : ADDPROG  BAK  : DOCFILE2 BAK
```

Enter an ambiguous file specificaition to delete several files at once. For example, the following sequence deletes all files with the primary filename NEWPROG, then all BAK files.

```
OA>ERA B:NEWPROG.*
OA>DIR B:
Directory for User 0
B: DOCFILE1 TXT : DOCFILE2 TXT  : DOCFILE3 TXT : ADDPROG ASM
B: DOCFILEl BAK : CHECKPRG ASNI : DOCFILE3 BAK : CHECKPRG BAK
B: ADDPROG BAK  : DOCFILE2 BAK

OA>ERA B:*.BAK

OA>DIR B:
Directory for User 0
B: DOCFILEl TXT : DOCFILE2 TXT : DOCFILE3 TXT : ADDPROG ASM
B: CHECKPRG ASX
```

The completely ambiguous filename *.* in an ERA command tail carries the most potential for disaster. Frequently, a user hoping to clear of f a data disk in drive B forgets to enter a disk specification or make B his current drive and finds he has erased his system disk by mistake. MP/M II queries any ERA *.* command, even if it contains a disk specification, to verify that you do indeed want the disk directory wiped clean. The sequence below illustrates such a.. exchange.

```
OA>DIR B:
Directory for 'iser 0
B: DOCFILE1 TXT   :  DOCFILE2 TXT    DOCFILE3 TXT   : NEWPROG ASM
B: NEWPROG BAI~   :  ADDPROG  ASM    DOCFILEI 13AK  : CHECKPRG ASM
B: DOCFILE3 BAF   :  CHECKPRG BAK    ADDPROG BAK    : DOCFILE2 BAK

OA>ERA *.*
Confirm delete all user files (Y/N)?N
OA>ERA B:*.*
Confirm delete all user files (Y/N)?Y

OA>DIR B:
Directory for User 0
File not Fouric

OA>DIR
Directory for User 0
A:  DIR       PRL  : PIP       PRL :  SUBmIT    PRL :  ERAQ      PRL
A:  ED        PRL  : ASM       PRL :  DDT       PRL :  LOAD      PRL
A:  STAT      PPL  : TOD       PRL :  DUMP      PRL :  DUMP      ASM
A:  TEST1     ASM  : TEST2     ASM :  DEBLOCK   ASM :  DISKDEF   LIB
OA>PHWEW!
PHWE!?

OA>
```

Note that ERA cannot delete files from a Read Only drive or a Read Only protected disk.

To erase password protected files, include the password as part of the file specification in the command line. If the password is missing, ERA prompts you for it. You can not see the password on the screen when you type it.

The [XFCB] option of the ERA command erases only the Extended File Control Blocks for the file specified by filespec. An XFCB contains password and time stamping information for a file. The ERA [XFCB] option is useful in reclaiming disk space that was used for time stamps and/or passwords, assuming the time stamps or passwords are no longer needed. (XFCBs are discussed in section 3, File Specifications, and in Section 7, The SET Command).

## 9.3 The ERAQ Command

**Syntax:**

```
ERAQ filespec
ERAQ filespec[XFCB]
```

The ERAQ command i,haves exactly like the ERA command with the addition of a query before each erasure. The following example illustrates the use of the EPAQ command.

```
IA>ERAQ C:*.PRL
00:12:27 A:ERA~       PRL    (USER 0)
C:ASM          PRL ?y
C:CONSOLE      PRL ?n
C:DIR          PRL ?y
C:DSKRESET     PRL ?y
IA>
```

In the above example, the current user number is I and the drive is A. The day-file option is on and the time is 12:27. The system found the ERAQ.PRL program on drive A, User 0. Since MP/M II was able to find ERAQ in User 0, we know that ERAQ must have an attribute of SYS. We also know that you do not need a password to READ the file. The *.PRL files are on drive C, User 1. In the example, the user instructs ERAQ to delete all the files except C:CONSOLE.PRL.

The [XFCB] option of the ERAQ command erases only the Extended File Control Blocks for the file specified by filespec. This is useful in reclaiming space on the disk used by time stamping options, assuming they are no longer needed.

## 9.4 The REN Command

Syntax:

```
REN newname.typ = oldname.typ
REN newname.typ = d:oldname.typ
REN newname.typ = d:oldname.typ;password
```

REN, MP/M II's rename command, replaces an old filename with a new filename in the disk directory. It can change either the primary filename or the filetype or both. The REN command accepts an ambiguous filename or filetype. REN used with a password transfers the password from the source file to the destination file.

Add a drive specification to the old or new filename if the file to be renamed is not on your default drive. If you include a drive for both the destination and source files, you must designate the same drive in both file specifications.

Here are some examples of the REN command:

```
OA>DIR B:
Directory for User 0
B: DOCFILE1 TEX : DOCFILE2 TEX : DOCFILE3 TEX : NEWPROG ASM
B: NEWPROG BA:~ : ADDPROG  ASM : DOCFILEl BAK : CHECKPRG ASM
B: DOCFILE3 BAK : CHECKPRG BAK : ADDPROG  BAK : DOCFILE2 BAK

OA>REN B:SECTION?.TEX=DOCFILE?.TEX

SECTION1.TEX=DOCFILEl.TEX
SECTION2.'rEX=DOCFILE2.TEX
SECTION31.TEX=DCCFILE3.TEX

OA>DIR B:
Directory for User 0
B: SECTION1 TEX : SECTION2 TEX : SECTION3 TEX : NEWPROG ASM
B: NEWPROG  BAK : ADDPROG  ASM : DOCFILEl BAK : CHECKPRG ASM
B: DOCFILE3 BAK : CHECKPRG BAK : ADDPROG  BAK : DOCFILE2 BAK

OA>REN B:DOCFILEl.TEX = SECTION1.TEX OA>REN
DOCFILE2.TEX=B:SECTION2.TEX
OA>REN B:DOCFILE3.TEX=B:SECTION3.TEX OA>DIR B:
Directory for User 0
B: DOCFILEI TEX : DOCFILE2 TEX : DOCFILE3 TEX : NEWPROG ASM
B: NEWPROG  BAK : ADDPROG  ASM : DOCFILE1 BAK : CIIECKPRG ASM
B: DOCFILE3 BAK : CHECKPRG BAK : ADDPROG  BAK : DOCFTLE2 BAK
```

If the file you want to rename is password protected and you do not include the password in the command line, REN prompts you for the password. The password does not show on the screen when you type it in.

If you enter a new filename that is already in the directory, REN returns a "Not renamed" error message. if you enter an old filename that is not listed in the file directory, REN returns a "No such file to rename" error message. For example:

```
OA>REN B:ADDPROG.ASM=ADDPLUS.ASM
Not renamed: B:ADDPROG.ASM already exists, delete (YIN)?

OA>REN B:ADDPLUS.ASM=ADDPORG.ASM
 No such file to rename
```

The REN command accepts wildcard filenames or filetypes, provided that the wildcard portions of the source and destination specifications match. The example below shows how to rename all of the files of type WRK to files of type TEX.

```
OA>REN *.TEX=*.WRK
```

# Section 10
# TOD, SCHED, PRINTER, SPOOL, STOPSPLR, SUBMIT

### 10.1 The TOD Command

Syntax:

```
TOD
TOD PERPETUAL
TOD mm/dd/yy hh:mm:ss
```

The TOD command sets and displays the system time-of-day. TOD with no argument returns the system date and time. The TOD PERPETUAL command specifies a continuous display of the date and time ("perpetual" can be abbreviated "p"). This display can be stopped by striking any key on the console.

TOD with the date and time supplied sets the date and the time to the hours, minutes and seconds specified. This TOD command allows you set the exact time by striking any key.

The following examples illustrate the TOD command.

```
0A>TOD<cr>
Wed 07/08/81 01:18:24

0A>TOD 07/13/81 16:40:50<cr>
Strike key to set time K
Mon 07/13/81 16:40:50 0A>
```

MP/M II displays the date and time as zero until you use the TOD command to set them. Note that when you power down or reset the entire system, you must reset the date and time.

### 10.2 The SCHED Command

Syntax

```
SCHED mm/dd/yy hh:mm command line
```

SCHED, when executed, waits in memory for the time and date to match the time-o-fday arguments supplied. When the month, day, year, hour, and minute specified matches the system date and time, SCHED automatically executes the program specified by command line. For SCHED to execute properly, it must have the SCHED.BRS and SCHED.RSP resident in memory, and SCHED.PRL on the disk.

When using SCHED, you must be sure the date and time are set correctly with the TOD command. Note that you must reset the date and time whenever the system is turned off or rebooted.

## 10.3 The PRINTER Command

Syntax:

```
PRINTER
PRINTER n
```

The printer command displays or sets the printer used by a particular console. Several consoles can share the same printer, but only one process can use the printer at a time. MP/M II expects the first printer assigned to the system to be printer number 0. The second printer is printer 1, and so on. The printer number is specified by n. When the PRINTER command includes an n, PRINTER sets the list device to printer number n. When you enter PRINTER without the n option, the system returns the number of the printer currently assigned to your console.

The simplest method of using the printer is to enter a ^P. From that point on, anything displayed on your console will also be printed on the printer. If a ^P is entered While the printer is printing, the console displays a message that the printer is busy

The following example shows the use of the PRINTER command.

**OA>PRINTER**
List Number = I

**OA>PRINTER 2**
List Number = 2

OA>^P

Printer busy.

## 10.4 The SPOOL Command

syntax:

```
SPOOL filespec
SPOOL filespec, filespec...
SPOOL filespec [DELETE]
```

SPOOL sends the file or files specified by filespec to the printer, leaving the console free for further input. SPOOL with the [DELETE] option deletes the file after spooling. Thc SPOOL command invoked from a file of type PRL on disk differs sightly in performance from SPOOL invoked as a Resident System Process (RSP).
When SPOOL is a Resident System Process, it displays no message.
When you invoke the SPOOL.PRL command, the console displays the following message.

```
OA>SPOOL
MP/M II V2.0 Spooler
 Enter STOPSPLR to abort the spooler.
 Enter ATTACH SPOOL to reattach console to spooler
***Spooler detaching from console***
OA>
```

If the SPOOL program is a PRL file, it can be invoked by as many users as the available number of memory segments allows. The command tail can contain up to ninety four characters.

If SPOOL is a Resident System Process, a maximum of three SPOOL commands can be entered, whether by one user or three different users. The command tail for an RSP can only be sixty characters long.

## 10.5 The STOPSPLR Command

Syntax:

    STOPSPLR

The STOPSPLR utility stops the spooling operation in progress and empties the spool queue. If you enter a STOPSPLR command while SPOOL is idle, the console displays the following message:

    Spooler not running

STOPSPLR with a console number substituted for the optional n stops a currently executing SPOOL.PRL command initiated from console

If the SPOOL operation is a Resident System Process, the STOPSPLR command invoked at ANY console aborts the current spooling operation in progress.

## 10.6 The SUBMIT Command

Generally, MP/M II accepts one command line at a time. However, if it is necessary to enter the same sequence of commands several times, it is more efficient to submit the commands as a group for sequential execution. MP/M 11's SUBMIT command directs the entry and execution of a sequence of MP/M II commands. To use SUBMIT, you must first create a file with a SUB filetype. The SUB file contains MP/M I commands for the SUBMIT program to execute. Section 10.6.1 details the creation and contents of the SUB file.

The SUBMIT command line includes the SUB filename and optional parameters. In response to the command, SUBMIT creates a file $n$.SUB that contains the commands listed in filename.SUB integrated with any parameters from the command line. MP/M II then executes the commands from this file rather than from user commands at the keyboard. Section 10.6.2 elaborates on the operation of the

SUBMIT program. During execution, MP/M II monitors the keyboard and the SUB file for certain conditions to abort the process. Section 10.6.3 discusses aborting SUBMIT and Section 10.6.4 describes the new INCLUDE submit command option.

### 10.6.1 Creating the SUB File

SUBMIT requires that the group of commands to be executed be submitted in a file with a filetype of SUB. Therefore, to use the SUBMIT program, you must first create a file with a SUB filetype. Create the SUB file as you would any text file, by using MP/M II's editor, ED, or any other editor.

In the editor enter the commands in the order You want SUBMIT to execute them. Enter only one command per line. For example, a submit file START.SUB may include the following commands:

```
STAT B:*.*
ERA B:*.BAK
DIR B:
PIP B:=A:TEX.COM
```

When executed, this list of commands reports on the size and attributes of files, and available space on drive B, then erases all backup files from the disk, and displays the directory to verify the erasures. Finally, the system copies the program Tex to the B drive.

A submit file can contain any combination of commands. it another SUBMIT command is part of the submit file, that command must be the last in the sequence. This is called chaining submits. Another method of including additional submit commands in a submit file is the $INCLUDE submit option described in Section 10.6.4.

It may be useful to include "formal parameters" or place holders in your submit file if you need different parameter values each time the submitted commands are executed. For example, you might want to specify the destination drive or a certain file to be erased or copied. A dollar sign, $, followed by an integer denotes a formal parameter. $1 denotes the first formal parameter; any subsequent place holders are denoted by $2, $3, ... $n. In the following example, START.SUB shown above is modified to contain two formal parameters.

```
STAT $i:*.*
ERA $1:*.BAK
DIR $1:
PIP $l:=A:$2.COM
```

In this example, $1 is .: formal parameter for a drive specification. $2 is a formal parameter for E. primary filename. When you enter c, SUBMIT command, the system pairs any actual parameters in your SUBMIT command line with the formal parameters in the SUB command file. The actual parameters in the command line are the values you want the system to substitute for $1...$n everywhere in the SUB file. The SUBMIT command below substitutes B for $1 and TEX for $2 in the START.SUB file shown above.

**OA>SUBMIT START B TEX**

Because the single dollar sign indicates a formal parameter, use two dollar signs, $$, to include a normal dollar sign in the submit file. SUBMIT reduces $$ to a single dollar sign.

Comments and operator instructions are not executable, but they are useful for others who may read your submit file. To include a comment in the submit file, begin its input line with a semicolon. MP/M II echoes all comments at the console. The submit file START.SUB with comments looks like this:

```
STAT $1:*.*
ERA $1:*.BAK
; THIS ERASES ONLY BACKUP FILES.
DlR $1:
; CHECK DIRECTORY TO VERIFY ERASURES.
PIP $l:=A:$2.COM
```

### 10.6.2 Operation of SUBMIT

Syntax:

```
SUBMIT filespec
SUBMIT filespec $1 $2 $3
```

After you create the SUB file of MP/M II commands, use SUBMIT to execute all the commands sequentially. As with all MP/M II commands, you must first enter tile command keyword following the MP/M II prompt. The submit filename follows the keyword. To submit START.SUB, enter the following command 'Line to the MP/M II prompt

**OA>SUBMIT START**

To access a submit file on an alternate drive, precede the filename with a drive specification as shown below.

**OA>SUBMIT B:START**

If the SUB file contains formal parameters, for example $!,$2, the command line must also include a command tail specifying the actual parameters to substitute for the formal parameters. For example, if you submit the START.SUB file below,

```
STAT $1:*.*
ERA $1:*.BAK
DIR $1:
PIP $1:=A:$2.COM
```

the command line must specify two actual parameters to substitute for $1 and $2. Note that in this case a colon, :, follows each occurrence of the formal parameter $1 in the submit file. The following command line includes two actual parameters: a valid drive name and primary filename.

**OA>SUBMIT START B TEX**

For SUBMIT to combine the commands from the SUB file with the actual parameters from the command line, it creates a temporary file. In response to the above command, SUBMIT creates a file $n$ . SUB on the temporary file disk (designated when MP/M II is generated). MP/M II assigns a number to n to keep track of more than one submit file at a time. For the above example, the $n$.SUB file looks like this:

```
STAT B:*.*
ERA B:*.BAK
DIR B:
PIP B:=A:TEX.COM
```

SUBMIT inserts "B" every time the first formal parameter, $1, appears in START.SUB, and inserts "TEX" every time the second formal parameter, $2, appears in START.SUB. (The $n$.SUB file cannot be typed at the console as it is in a special format).

If the number of parameters in the submit file is greater than the number in the command line, nothing is substituted for the extra parameters. If more parameters are entered in the command line than exist in the file, the extra parameters are ignored.

After creating the $n$.SUB file, the SUBMIT program terminates, and MP/M 17 regains control. When in control, MP/M 11 checks the temporary file disk for a $n$.SUB file. If $n$.SUB exists, MP/M II executes the commands in this file. As SUBMIT processes $n$.SUB, each command is displayed at the console. The console looks just as ' does when you enter the commands manually one at a time. After processing $n$.SUB, the system erases the file and returns control to MP/M II.

### 10.6.3 Aborting SUBMIT

Any error in the command line causes SUBMIT to display an error message, abort that program, and proceed to the next SUBMIT command.

You can abort SUBMIT by entering a ^C at the console. The system responds with the prompt:

```
Terminate Submit?
```

If you enter a Y the system aborts the SUBMIT operation but not the current process. After your response to this question, a second question is displayed:

```
ABORT   (Y/N) ?
```

If a Y is entered at this point, the currently executing process is aborted. If an N was entered following the "Terminate Submit?" prompt, the remaining commands in the submit file continue to execute.

If *SUBMIT* cannot find the *SUB* file specified in the command line, it displays the following message. The line number of the submit file at which the error occurred is substituted for nnn.

```
Error on line nnn no 'SUB' file present.
```

#### 10.6.4 The INCLUDE Submit Option

MP/M II SUBMIT introduces an include facility that allows you to nest up to four levels of submit files. The INCLUDE command is placed on a line in your submit file and has the following syntax:

```
$INCLUDE submitfilename
$INCLUDE submitfilename $1 $2 $3 ...
```

The $INCLUDE is followed by the name of a submit file of type SUB. The "SUB" is assumed and can be left out.

The INCLUDE 1~Ie is replaced by the entire submit file named submitfilename. If the included submit file requires parameters they can be included in the $INCLUDE line as shown in the second syntax line above. The following is an example of the $INCLUDE submit option.

```
Content of SUB1.SUB            Content of SUB2.SUB

STAT $i:*.*                    DIR F:$l
ERA $1:*.BAi~                  ERA F:$l
$INCLUDE SUB2.SUB *.BAK        DIR F:
DiR $1:
PIP $l:=A:$2.COM
```

# Section I I
# The PIP Command

## 11.1 Introduction to PIP

PIP stands for Peripheral Interchange Program. At the user's request, PIP copies files from one, peripheral or location to another. Usually this involves copying a file from one disk or user to another disk or user. For example, PIP can read a PRE, file on drive A and make a copy of it on drive B. PIP can also concatenate or join two or more files from the same disk and copy them into one file on another disk, and even rename the new file in the same process. The details of using PIP on disk files are given in Section 11.2. Section 11.3 discusses PIP and password protected files.

PIP can also copy a file to a peripheral other than a disk, such as a printer or console. Section 11.4 tells how to use PIP with a peripheral device.

PIP can also perform certain operations as it makes a new copy of a file. For example, PIP can translate uppercase letters in the original file to lowercase in the new file. These operations and the PIP options that initiate them are described in Section 11.5.

PIP is not an interactive program, but does have a repertory of console messages that inform the user of copy status and error conditions. Section 11.6 summarizes PIP console messages.

To invoke PIP, enter its name as a command to the MP/M 11 prompt. PIP can accept an optional command tail, as shown in the syntax lines below:

Syntax:

```
PIP
PIP destination filespec [Gn] = source filespec [option list]
```

It should be noted that either the destination drive or the destination filename is optional, but NOT BOTH. One or the other must be present. The same is true of the source drive and filename. It should also be noted that the source drive and filename cannot be exactly the same as the destination drive and filename. Either the filename or the drive must be different. There is, of course, one exception to this rule. If the default or specified user number in the source file specification is different from the default or specified user number in the destination file specification, then the source and destination drives and filenames can be exactly the same.

If you don't enter a command tail after the command keyword, PIP assumes you want to enter multiple command tails and responds with the prompt, *. When you enter a command tail and carriage return to this prompt, PIP performs the task you requested and reissues the * prompt To return to MP/M II, simply press the carriage return key. The sequence below demonstrates PIP accepting multiple commands.

```
OA>PIP
MP/M II PIP VERSION 2.0
*B:APPl.BAK =  A:APPI.ASM
*B:ASM.PRL = A:ASM.PRL
*<cr>
OA>
```

If you enter a command tail directly after the command name, PIP performs the requested task and returns immediately to MP/M II without issuing the * prompt.

The destination in the command tail is the file or peripheral device that is to receive the data. The source can be one or more files or devices to :~e copied to the destination. If multiple sources are entered, with a single destination PIP copies them to the destination in the order they appear in the command line, from left to right. In this case, the destination must include a filename into which the multiple source files are to be concatenated. Each source must be separated from the next by a comma. A space between the comma and the next source is optional

PIP options, shown in the syntax line as [option list], must immediately follow t:,~ source file specification on which PIP operations are to be performed.

If you enter a PIP command that specifies a device as a destination, you can abort the copy operation by pressing any key on the keyboard, or by using the ^C. in fact, you can always abort PIP with a ^C from the console. if PIP is detached and the console is in use, use the ABORT command from another console.

Source files are opened in Read Only mode to permit other users to type a file while it is being copied. If a physical error occurs during a PIP copy operation, the operation is aborted and an appropriate error message displayed. if  file cannot be copied because it is locked or password protected, PIP displays En appropriate error message, skips the file, and continues the copy operation. (See the MP/M II Programer's Guide for a discussion of locked and unlocked files).

## 11.2 PIP and Disk Files

PIP is most frequently used to make a copy of a disk file on another disk. PIP accepts ambiguous filenames and other options in its command tail. It can also concatenate or join multiple source files from a disk, making one longer destination file on another disk.

PIP can copy ASCII files, such as program source files, or non-ASCII files, such as memory image PRL files. PIP expects an ASCII file to end in ^Z , which ED and other file editors insert automatically. PIP does not expect any special character at the end of a non-ASCII file.

Note: If the number of characters in an ASCII file is an exact multiple of 128, the file will not be terminated with a ^Z.

in a disk film transfer, both the destination and source in PIP's command tail are file specifications. For example:

**OA>PIP B:DDT.PRL = A:DDT.PRL**

If you do not specify a drive letter as part of the filename, PIP assumes the file is on the default drive.

To execute your command , PIP first checks to see if a file with the same name as the destination file already exists on the destination disk. If there is a file of the same name on the destination disk, PIP checks to be sure the file can be opened. if for some reason the file cannot be opened, perhaps because it is already being accessed by another user, PIP displays an error message and aborts. If no file of the same name exists, or it the file can be opened, PIP creates a temporary file of type $$$ on the destination disk. The temporary destination file has the same primary filename as the destination file of the command tail, but its filetype is .$$$. If PIP aborts for some reason, you must delete the .$$$ file that remains on your disk.

After creating the temporary destination file, PIP begins copying the source tile or files into the temporary file. When the copying operation is complete, PIP deletes any file on the destination disk that has the same name as the destination file specified in the command tail.

If the file with the same name has an attribute of Read only, PIP displays the following prompt on the console asking whether or not to delete the old file.

```
File Read Only Delete Y/N ?
```

If you enter an "N" , the copy operation aborts and nothing is changed. (The W option instructs PIP to write over files with the Read Only attribute. See Section 11.4). If you enter a "Y", PIP deletes the Read Only file and renames the temporary file with the filename given in the destination of the command tail.

PIP allows you to abbreviate file specifications in the command tail. In certain instances, it is not necessary to enter a complete file specification as the source or destination; sometimes only a drive specification is required. For example, if you want the destination file to have the same name as the source file, you need only enter the destination drive specification and the complete source file specification as shown below:

**OA>PIP B:=A:TOD.PRL**

This kind of abbreviation also works if you specify the destination file specification and only a drive specification for the source. PIP then searches the source disk for a file whose name matches the destination specification, and if it finds one, copies it to the destination disk. Here is an example of such a command:

**OA>PIP B:TOD.PRL = A:**

Note that the file specification can be abbreviated even further if the source or destination it specifies is on your default drive. For example, the first command below makes a copy of TOD.PRL from the default drive A to the destination drive, B. The second command copies the file APP1.ASM from drive B to the default drive

**OA>PIP B:=TOD.PRL OA>PIP APPI.ASM = B:**

When you give an ambiguous fi1e name as the source specification, PIP cpies all files matching the ambiguous specification to the destination drive. It adds the copies to the destination disk's directory, giving them the same primary filenames, filetypes, passwords and attributes as they had on the source disk. As it does this, PIP lists each filename that satisfies the ambiguous reference at the console. This is handy for transferring PRL files from the distribution disk to your new, system disk, as shown in the following example.

```
OA>DIR
05:27:54 A:DIR           PRL    (User 0)
Directory for User 0:
A: BNKBDOS    SPR  :  BNKXDOS   SPR : RESBDOS    SPR  : TMP        SPR
A: XDOS       SPR  :  ABORT     RSP : MPMSTAT    RSP  : SCHED      RSP
A: SPOOL      RSP  :  MPMSTAT   BRS : SCHED      BRS  : SPOOL      BRS
A: ABORT      PRL  : ASM        PRL : CONSOLE    PRL  : DIR        PRL
A: DSKRESET   PRL  : DUMP       PRL : ED         PRL : ERA         PRL
A: ERAQ       PRL  : MPMSTAT    PRL : PIP        PRL : PRINTER     PRL
OA>DIR B:
05:28:14 A:DIR           PRL    (User 0)

Directory for User       0:
File not Found

OA>PIP B:=A:*.PRL
05:28:30 A:DIR           PRL    (User 0)
Copying
ABORT.PRL
ASM.PRL

OA>DIR B:
05:32:04 A:DIR           PRL    (User 0)

Directory for User 0:
B:  ABORT     PRL  : ASM        PRL  : CONSOLE   PRL  : DIR        PRL
B:  DSKRESET  PRL  : DUMP       PRL  : ED        PRL  : ERA        PRL
B:  ERAQ      PRL  : MPMSTAT    PRL  : PIP       PRL  : PRINTER    PRL
```

To concatenate several disk files into one large file, enter a string of file specifications as the source in the command tail. Multiple source files must be in ASCII format unless specifically overridden by the 0 parameter. They are written into the destination file in the order they appear in the command tail, from left to right. For example:

**OA>PIP DESTINAT.TXT=SOURCE1.TXT,SOURCE2.TXT,SOURCE3.TXT**

In a concatenation command, the destination filename and filetype can also appear as the source filename and filetype. This is possible because PIP Copies Source files into a temporary file. When the operation is complete, PIP checks the, directory for duplicate destination filenames, deletes any it finds, and renames the temporary file with the specified destination name.

## 11.3 PIP and Passwords

If a file is password protected, the password must be entered in the command line following the file to which it belongs. The filename is separated from the password by a semicolon. Consider the password as part of the filename when placing drive specifications, passwords and options in command lines. Only one password can be specified for ambiguous copy operations where any matching files are password protected. When a password is required, PIP tries the password. MP/M II tries the default password. If the password fails, the file is skipped and the failure noted.

When a destination filename is specified with E~ password, that password is assigned to the destination file. When a destination filename is specified with no password, no password is assigned to the destination file. When only the destination drive is specified, the destination file receives the password that is assigned to the source file. If the source file has no password, the destination file receives no password.

File attributes .,re copied with the file. This includes the Read Write or Read Only and SYS or DIR file attributes and the user defined attributes Fl through F4.

Some password protection modes are also copied with the files. When a destination .le is specified with a new password, the password protection mode is automatically set to READ. This means that a password is required to read the file. (See the SET command in Section 7). When the destination drive with no filename is specified, and the file and password are being copied from the source, the destination file receives the same password protection mode as the source file.

## 11.4 PIP and Other Peripheral Devices

In general, PIP treats a peripheral device in the same way it treats a file. Filenames and device names can be used as source or destination names in ~ PIP command tail.

A device specified as a source in a, PIP command tail must be capable of transmitting data. For example, a console is a valid source in a PIP command tail, but a printer is not. Note that PIP expects the same end-of-file signal from a source peripheral device as from a source disk file. ASCII files must end in ^Z. Non-ASCII files terminate in a physical end-of-file (EOF).

You can use the console as a source device in a PIP command. You must terminate, each line, of text with a carriage return AND line feed keystroke. Type a ^Z to exit the file and return to the system prompt.

A device specified in a PIP command tail can be a logical device or a special PP device. Table 111 defines the two MP/M 11 logical devices.

**Table 111. MP/M II Logical Devices**

| Name | Device |
|---|---|
| CON: | the console |
| LST: | a list device or printer |

Each logical device is assigned a physical device. You can specify any valid logical device as a source or destination in a PIP command tail. PIP then copies data to or from the physical device associated with that logical device. For example, to list the file APP.PRN on disk B: it the system printer, enter the PIP command shown below:

**OA>PIP LST:=B:APP.PRN**

PIP then copies the disk file to the physical device associated with LST:, generally a line printer.

Besides logical devices, PIP also allows special names to be included as devices in a command tail. In general, these special devices perform specifid tasks, such as adding an end-of-file character to the end of the destination file or adding leading or trailing zeroes to E~ file. Table 11-2 summarizes the special PIP devices.

**Table 112. Special PIP Devices**

| Name | Function |
|---|---|
| NUL: | A source device that sends 40 zeroes (nulls) to the destination device. |
| EOF: | A source device that sends an end-of-file character to the destination Device. PIP automatically adds a ^Z to the end of all ASCII data transfers. |
| PRN: | A destination device, that performs the following functions as data is copied: expands tabs to every eighth character position, numbers lLines, inserts an initial page eject and an additional page eject every 60 lines. (Same as LST: with [t8np]). |

## 11.5 PIP Options

PIP options enhance a copy operation. Generally, they request that PIP perform some optional process on the source or destination. Some PIP options described here expand tab characters, translate the case of alphabetic characters, extract portions of source files, and verify that copies are made accurately.

PIP options are local options following the name of the file specification to be affected during the copy. Options must be enclosed in square brackets. Certain options require an argument, which can be a number or character string. Within the brackets, multiple options can be separated by spaces. However, no space can separate an option letter from its argument. Table 113 describes the function of each PIP option, using n to represent a numeric argument and s to represent a string argument.

**Table 11-3. PIP Options**

| Letter | Function |
|---|---|
| A | To back: up only the files that have been modified since the last backup, use PIP with an ambiguous filename and the Archive option. PIP wit.. the [A] option copies only the files that have beer) modified. |
| Dn | Delete any characters pastcolumn n. This option followsa source filethat contains Lines too long to be handled by the destination device; for example, an 80 character printer or narrow console. The number n should be the maximum column width of the destination device. |
| E | Echo transfer at console. When this option follows ~i source name, PIP displays the source data & theconsole as thecopy is taking place. The source must contain ASCII data. |
| F | Filter form-feeds. When this option follows a source name, PIP removes all form-feeds Imbedded in the source data. To change form feeds set for one page length in the source File to another page length in the destination file, use , the F option to delete the old form feeds and the P option to simultaneously add new form-feeds to the destination file. |

**Table 11-3. (continued)**

| Letter | Function |
|---|---|
| Gn | This option is used to "Get from" or "Go to" the user number specified by n. The [Gn] option is the only valid option for destination files. When the [Gn] option is placed after the source filename, PIP looks for the source file in the user area specfied by n, on the currently logged or specified drive. When the [Gn] option is placed after the destination filename, PIP places the destination file under the user number specified by n on the designated drive. n must be a valid user number between 0 and 15. |
| H | Hex data transfer: PIP checks all data for proper Intel hexadecimal file format. This requires a file extension of HEX. Removes Non essential characters between hex records during the copy operation. The console displays a prompt for corrective action in case an error occurs. |
| I | ignore :00 records in the transfer of Intel hexadecimal format files. This requires a file extension of HEX. The I parameter automatically sets the H parameter. |
| K | Kill display option stops the display of filenames on the console during ambiguous copies. This permits the user to detach PIP from the console during long copy operations. |
| L | Translate uppercase alphabetic in the source file to lowercase in the destination file. This option follows the source device or filename. |
| N | Add line numbers to the destination file. When this option follows the source filename, PIP adds a line number to each line copied, starting with 1 and incrementing by one. The line number is followed by a colon. If N2 is specified, PIP adds leading zeroes to the line number and inserts a tab after the number. If the T parameter is also set, PIP expands the tab. |
| O | object file transfer for non-ASCII files. PIP ignores the physical end-of-file during concatenation and transfer. |

**Table 11-3. (continued)**

| Letter | Function |
|---|---|
| Pn | Set page length. n specifies the number of lines per page. When this option modifies a source file, PIP includes a page eject at the beginning of the destination file and at every n lines. If n = I or is not specified, PIP inserts page ejects every 60 lines. When the F option is also specified, PIP removes form feeds from the source data before inserting new form-feeds at the page length specified by n. |
| QS^Z | Quit copying from the source device after the string s. When used with the S parameter, this option can extract a portion of a source file. The string argument must be terminated by a ^Z. Note: When PIP is invoked without a command tail, i.e., "PIP <cr>", the character string is not translated to uppercase, otherwise it is. |
| R | Read system files. Normally, PIP cannot find files marked with the system attribute in the disk directory. But when this parameter follows a source filename, PIP can copy a system file, including its attributes, to the destination. |
| Ss^Z | Start copying from the source device at the String s. The string argument must be terminated by a Z. When used with the Q parameter, this option can extract a portion of a source file . Both start and quit strings are included, in the destination file. Note: When PIP is invoked without a command tail, i.e., "PIP **<CR>”,** the character string is not translated to uppercase, otherwise it is. |
| Tn | Expand tabs. When this option follows a. source filename PIP expands tab (^I) characters in the destination file PIP replaces each ^I. with enough spaces to position the next character in a column divisible by n. |
| U | Translate lowercase alphabetic characters in the source file to uppercase in the destination file. This option follows the source device or filename. |

**Table 11-3. (continued)**

| Letter | Function |
|---|---|
| V | Verify that data has been copied correctly. Whenthis option modifies the source name, PIP rereads the destination data after writing to checkfor copying errors. The destination must be a disk file. |
| W | Write over fileswith Read Only attribute. Normally, if a PIP command tail includes an RO fileFs a destination, PIP sends a query to the consoleto make sure the user wants to write over theexisting file. When this option follows a source name, PIP overwrites the RO file without aconsole exchange. If the command tail contains multiple source files, this option need follow only the last file in the list. |
| Z | Zero the parity bit. When this option follows a source name, PIP resets the parity bit of each data byte in the output. The source must contain ASCII data. |

The remainder of this section gives examples of PIP parameters and their effects. For example, when the command

**OA>PIP C0N:=A:WIDEFILE.ASM[D80]**

is executed, PIP truncates any source lines longer than 80 characters before sending them to the console device.

The command

**OA>PIP B:=A:LETTER.TEX[E1**

causes PIP to display the file LETTER.TEX line by line at the console as it is copied to the destination.

The following command specifies two parameters to change the number of lines printed on a page:

**OA>PIP LST:=B:LONGPAGE.TEX[F'P65]**

In response to this command, PIP removes any form-feeds it finds in the source data, then inserts a form-feed at the beginning of the printout arid after every sixty-fifth line. The form-feeds remain in their original position in the source file; only the destination has a new page length.

Here is another multiple-parameter command that enhances file printout:

**OA>PIP LST:=B:PROGRAM.ASM[N2T8U]**

When this command is executed, PIP numbers each line, inserts a tab between the number and the beginning of each line, expands those tabs and any others to every eighth column, and translates lowercase alphabetics to uppercase. These enhancements affect only the printout; the source file is unchanged.

The following command extracts part of a source file.

**OA>PIP P0RT10N.TEX=B:LETTER.TEX[SDear Sir,z QSincerely,zj**

In this case, the salutation, the body of the letter, arid the closing are transferred to the file PORTION.TEX. Any information before "Dear Sir," such as an address, or after the closing are not included in the destination file. PIP does not alter the source file.

The following example command would be useful to update a system disk with new versions of command files:

**OA>PIP B:=A:*.PRL[VWR]**

In this case, disk B is updated with the commands on disk A. The R parameter allows PIP to read any PRL files on disk A that ,re marked with the system. attribute. The W parameter allows PIP to write over any Read/Only files. The V parameter causes PIP to reread and verify that each file is copied correctly.

**OA>PIP F:[g3] = E:USERFILE.DAT**

This command causes the file USERFILE.DAT to be transferred from the current user area on rive E to user area 3 on drive F.

**OA>PIP G:=B:*.PRL(K]**

Option K instructs PIP to cancel the console listing of all the files it is copying with a filename extension of PRL. This enables the user to detach PIP from the console while it finishes the copy procedure.

**Oa>PIP B:=*.*[A]**

This command backs-up only the files on the default disk that have been edited or modified. The backup disk is placed in the B drive. After the backup is completed, each of the files on the default disk are marked as archived (the archive indicator is turned on) until the next time one, of them is modified.

## 11.6 PIP Console Messages

During a copy operation, PIP might need additional input from the user. In certain cases, PIP might inform the user of copy status or error conditions. The messages PIP can display at the console are described in Table 114, below. All error messages are prefaced by the word ERROR.

Note: In some errors, several of the error messages described below might be combined into one message.

**Table 11-4. PIP Console Messages**

| Message | Meaning |
| --- | --- |
| ALREADY EXISTS | The temporary file already exists on the destination disk. Erase or rename it. |
| BAD PARAMETER | PIP does n o t recognize a character i n c 1 u d e d in square brackets as a parameter. |
| CAN'T DELETE TEMP FILE | This message appears as part of another error message. |
| CHECKSUM ERROR | The HEX file being copied has an invalid checksum. |
| CLOSE FILE | An error occurred when closing a file. This message appears as part of another error message. |
| COPYING ... | PIP displays this message before listing filenames that match an ambiguous source specification at the console. |
| CORRECT ERROR, TYPE RETURN OR CTL-Z | PIP gives you a chance to correct an invalid HEX record when using the [H] option. Type in the correct record or enter a ^Z to terminate the copy. |

**Table 11-4. (continued)**

| Message | Meaning |
|---|---|
| DESTINATION IS R/0, DELETE (Y/N)? "NOT DELETED" | Destination file has Read Only attribute. The user can specify whether to delete the old RO file that has the same name as the specified destination (Y), or abandon the temporary file (N). If N is specified, PIP displays the "NOT DELETED" message and aborts the copy operation. |
| DISK READ | A bad sector on the source disk was read. (Can mean wrong density). |
| DISK WRITE | A bad sector error occurred while writing the destination file. (Can mean wrong density). |
| END OF FILE, ^Z, ? | PIP encountered an unexpected end of file during a HEX file transfer. |
| FCB CHECKSUM | The Destination FCB has been corrupted. (Can mean bad memory). |
| FILE NOT FOUND | PIP cannot find a source file. Check your command line. |
| INCOMPATIBLE MODE | The file was opened by another process in an incompatible mode. (Locked or Read Only). |
| INTERNAL LOCK LIMIT EXCEEDED | The maximum number of locked files specified at system generation has has been exceeded. |

**Table 11-4. (continued)**

| Message | Meaning |
|---|---|
| INVALID DESTINATION | An invalid destination device has been specified in the command tail. |
| INVALID DIGIT | An invalid hexadecimal digit was found in the HEX file being copied ([H] option). |
| INVALID DISK SELECT | The drive selected is not implemented on the system. |
| INVALID FILENAME | There is a wildcard character improperly placed in a filename or filetype. |
| INVALID FORMAT | The command line contains a syntax error. |
| INVALID PASSWORD | The wrong password was specified for a file or the password was omitted. |
| INVALID SEPARATOR | A file concatenation command line contains invalid characters between file specifications. |
| INVALID SOURCE | An invalid source device has been specified in the command tail. |
| INVALID USER NUMBER | The G parameter, has an argument less than 0 or greater than 15. |
| LIMIT EXCEEDED | The maximum number of open files specified at system generation has been exceeded. |

**Table 11-4. (continued)**

| Message | Meaning |
|---|---|
| MAKE FILE | An error occurred when creating a file. This message appears as part of another error message. |
| NO DIRECTORY SPACE | The destination directory is full. Erase some files or get a fresh disk. |
| NONRECOVERABLE | A disk contains an actual physical defeat. This message can also mean that the drive is set to the wrong density for the disk in it, the disk was inserted improperly, or the drive door was left open. |
| OPEN FILE | An error occurred when opening a file. This message appears as part of another error message. |
| PRINTER BUSY | The list device is already in use. |
| QUIT NOT FOUND | PIP cannot find the string argument to a Q parameter in the source file. |
| RECORD TOO LONG | A HEX record exceeds 80 characters in a file being copied with the [H] option. |
| REQUIRES MP/M II | PIP was invoked on another operating system besides MP/M II version 2.0. |
| R/O DISK | The destination drive is set to Read Only mode and PIP cannot write to it. |

**Table 11-4. (continued)**

| Mess | Meaning |
|---|---|
| R/O FILE | The destination file is set to Read Only and PIP cannot write it. |
| START NOT FOUND | PIP cannot find the string argument to an S parameter in the source file. |
| VERIFY | When copying with the V option, PIP found a difference when rereading the data just written and comparing it to the data in its memory buffer. |
| USER ABORTED | PIP received a keystroke from the console while copying data to or from a device. Start over. Check for undeleted .$$$_files. |

# Section 12
# ED, The MP/M 11 Editor

## 12.1 Introduction to ED

To do almost anything with a computer you need some way to enter data, some way to give the computer the information you want it to process. The programs most commonly used for this task are called "editors." They transfer your keystrokes at the console to memory or to a disk file. MP/M II's editor is named ED. Using ED, you can create and alter text files.

To create a file, enter the data through ED at your console. ED stores the data in memory, then at your command, writes the data into a temporary disk file. To change an existing file, ED copies the file from the disk to memory. There you can use ED's editing commands to insert delete, and replace data. When you finish editing, ED writes all the changed data and any unchanged data in the original file Lack out to the disk in the temporary file. ED keeps the original fil_ as a backup, and changes the filetype to .BAK. ED renames the temporary file, filename.$$$ to filename.typ, the resulting edited file. Figure 12-1 shows the relationship of the original, temporary and backup files to the user and the memory buffer. The letters shown in parentheses are the ED commands that move data from one Location to another.

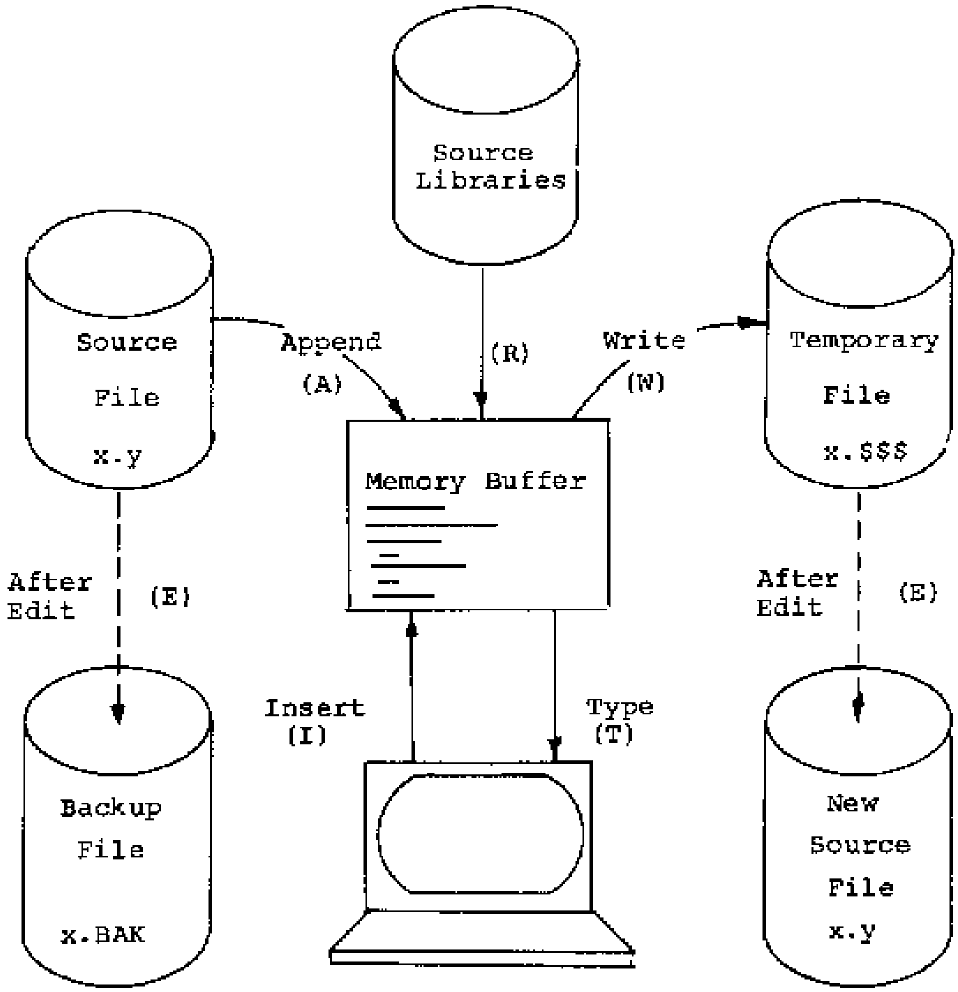


**Figure 12-1. ED's Input and Output Files and the Memory Buffer**

To edit a large file ED first copies a section of the file into the buffer. After editing that section, the user instructs ED to write out the edited section to the new file. Then at the user's command, ED copies another section of the large file into the memory buffer for the user to edit. For this reason, editing with ED is generally a one-direction process: you start at the beginning of the file and edit through to the end. Although you may edit backwards through the memory buffer, you may not reedit any sections ED has already written out to the new file without starting afresh at the top of the new file. How ED keeps track of the original file, the new file, and the data in the memory buffer is described in Section 12.2, ED Concepts and operation.

ED is a line-oriented character editor; that is, it treats a file as F. long chain of characters. Characters are grouped together in lines. ED defines a line as any characters after a carriage-return/linefeed sequence (<cr><lf>) up to and including the next <cr><lf>. (When you press the carriage-return key in insert mode, the linefeed is inserted automatically.)

Section 12.3 gives a quick introduction of how to get started with ED. Section .12.4 gives complete descriptions of all the ED commands. Section 12.5 describes ED's error messages.

## 12.2 ED Concepts and Operation

To keep track of editing progress through the file, ED uses four pointers: the source pointer, the memory pointer, the temporary pointer, and the character pointer. Each pointer marks a location where data can be manipulated. The source, memory, and temporary pointers keep track of where blocks of data can be moved. The character pointer resides in the memory buffer and indicates where editing can take place.

The source pointer (SP) always points to the next character in the original file that should be read into the buffer. When editing begins, the SP points at the first character in the file. After ED reads a section into the buffer, the SP points to the next character to be read into the buffer.

The memory pointer (MP) always points to the last character in the memory buffer. When ED copies data from the original disk file into the buffer, it ;dds the new data after the MP. After the copy operation is complete, ED moves the MP to indicate the new last character in the buffer. This prevents ED from overwriting data already in the buffer.

The temporary pointer (TP) always points to the next free location in the new disk file. ED writes data to the new file in the area after the TP. Then, when the write operation is completed, ED moves the TP to the end of the data it has just written in the file, that is ED moves the TP to the new next free location. This prevents ED from overwriting data already stored in the new file. Figure 122 shows the locations of the SP, MP, TP, and CP in the original file, memory buffer, and the new file.

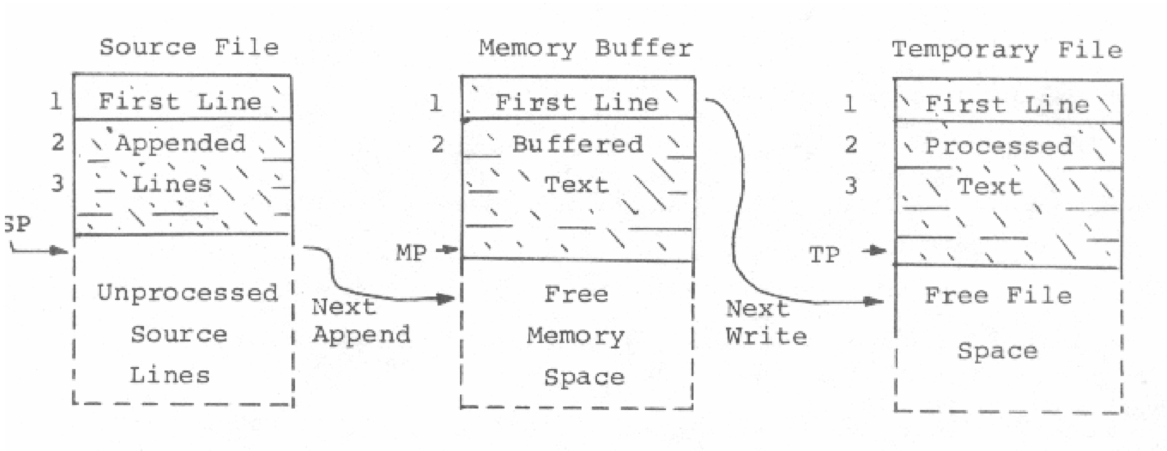


Figure 12-2. **ED's Pointers**

The character pointer (CP) indicates where editing can take place. The CP is always in the memory buffer, and can move anywhere within the buffer at the user's command. Imagine that the CP is always between two characters. An ED command with a positive numeric argument affects the characters after the CP (towards the bottom of the file) . ' ~,,i ED command with a negative numeric argument affects the characters before CP (towards the top of the file). ED is sometimes called a context editor because it makes changes in the context of or in relation to the CP. Section 12.3 introduces commands that move the CP, while Sections 12.4.3 and 12.4.4 give more detail. Table '.)I shows how a few ED commands move the CP through the buffer. In this table, the location of the CP in memory is represented by the special character ^.

**Table 121. Moving the Character Pointer**

| ED Command | Effect | Resulting Memory Buffer |
|---|---|---|
| #a | move contents of file Into the buffer | ^NOW IS THE TIME FOR ALL GOOD MEN |
| 11c | move CP 11 characters | NOW IS THE TIME ^FOR ALL GOOD MEN |
| 1 | move CP down one line | NOW IS THE TIME FOR ALL ^GOOD MEN |

When invoked, ED first allocates as much of memory as possible for the buffer. It then searches the disk directory for the filename given as an argument in its command tail. If it does not find a, matching filename, ED displays tine message NEW FILE and creates a temporary file, filename.$$$. If ED does find the file name, it opens that file for editing. Then ED creates temporary file with the extension .$$$, which will become the new file at the end of the, editing session. Finally, ED displays its prompt, *, at the console. To ED's prompt you can enter commands that copy characters from the original file to the memory buffer, move the CP through the buffer, display characters at the console, change characters, and write the edited characters out to the temporary file.

When you tell ED you have finished editing, ED writes any characters remaining in the buffer or uncopied from the original file into the temporary file and closes both files. Then ED deletes any backup file that may exist and renames the original file with the extension BAK. Finally, ED gives the temporary file the original filename specified in the ED command tail.

## 12.3 Starting with ED

Before starting an editing session, remember that ED requires a certain amount of disk space to perform its editing task. A general rule is to make sure that the unused portion of your disk is at least as big as the file you are editing  larger if you plan to add characters to the fi1e. When ED finds a disk or directory full, it abandons the edited version of the file. To check the amount of space on your disk, use the STAT command described in Section 6.

To invoke ED, enter its name to the MP/M II prompt as a transient command. The command ED must be followed by a specific file reference, one that contains no wildcard characters. The file specification identifies the file to be edited or created. For example:

**OA>ED LETTER.TXT**

The filename may be preceded by a disk specification, but a disk specification _s unnecessary if the file to be edited is on your default disk. if the file to be edited is password protected, ED prompts the console for a password. optionally, the file specification may we followed by a disk specification. In that case, ED creates the temporary file on the disk specified after the filename. For example:

**0A>ED LETTER.TXT B:**

In response t this command, ED opens the original file on drive A and creates a temporary file LETTER.$$$ on drive B. At the end of the editing session, ED renames the orginal file on drive A to LETTER.BAK and renames LETTER.$$$ on drive B to LETTER.TEX. There is a side effect to this command: the drive specified at the end of an ED command tail becomes the MP/M Ii default disk at the termination of the editing session. So, at the end of the editing session initiated by the command above, MP/M II returns the prompt:

OB>

If the file referenced in ED's command tail does not exist, ED responds with the following display:

```
OA>ED LETTER.TXT

NEW FILE
   : *
```

To add new text to the file, use an Insert (i) command as shown below. When you have finished inserting, type a ^Z to exit insert mode.

```
 : * i
1: Insert text ...
2: and more text ...
3: end of text ^Z
```

If the file specified in ED's command tail does exist, ED responds with a, similar display. To read the original file into the memory buffer, use an Append (a) command. In the example below, the pound sign, #, preceding the a command reads the whole file into memory

```
A>ED B:LETTER.TXT
       : *#a
      1: *
```

You can display lines of text simply by hitting the carriage return k c., y . A <cr> moves the CP to the beginning of the next line and displays the 'Line. if you reach the bottom of the buffer, you can return to the top by entering a Beginning (b) command, as shown below:

```
1: *<cr>
2:  and mote text...
2: *<cr>
3:  end of Lext.
3: *<cr>
 : *b
1: *
```

Sections 12.4.3 and 12.4.4 tell how to display the buffer's contents at the console by pages or line numbers and move the CP within lines.

To delete a line, use the Kill (k) command, ED automatically renumbers lines in the buffer when one is taken out, as shown below:

```
2: and more text
2: *k
2: *b
1: *<cr>
2: end of text
2: *
```

Section 12.4.1 discusses line numbers in the memory buffer. Section 12.4.5 tells how to delete individual characters as well as lines. Section 12.4.6 describes how ED searches for and replaces strings, and Section 12.4.8 explains how to move text blocks. Section 12.4.9 tells how to end an edit and exit ED. An Exit (e) command saves the contents of the buffer and returns control to MP/M II.

```
        2: *e

    OA>
```

## 12.4 ED Commands

In general, an ED command consists of a numeric argument and a command letter. Certain commands consist of a command letter and a character string followed by a carriage return or ^Z. Here is an example of each form:

```
23T
1OFexample^Z
```

The numeric argument in the first form is optional. If it is omitted, ED assumes an argument of one. Use a negative argument if the command is to be executed backwards through the memory buffer

ED accepts 0 as a numeric argument only in certain commands. In some cases, 0 causes the command to be executed approximately half the possible number of times, while in other cases it prevents the movement of the CP. The 0 option is described in detail in the definitions of the A, W, L, and P commands.

In some ED commands, it makes a difference whether you enter the command letter in upper or lowercase. When you enter these command letters in uppercase, you in effect press an internal caps-lock key. For example, if you enter an insert command as a capital I, all the characters you insert are translated to uppercase. Although the console may echo the characters in lowercase as you enter them, ED internally translates them and shows them in uppercase in the next display. The 1, F, S, and N commands have this translation feature, so see the description of these commands for more detail.

ED can accept character strings as a part of certain commands. In these command lines, the character string appears after the command letter and may be terminated by either a carriage return or a T Z When an ED command contains a character string, the string may not exceed 100 characters. Examples of character strings in ED commands are given in the descriptions of the F, S, N, I , and J commands.

When you are just starting to use ED, you will probably enter only one command at a time. However, ED is easier to use if you combine commands on a single command line. For example, when you make a change, you might want ED to display the corrected line so you can verify that the change was made correctly. It saves keystrokes and editing time to combine the editing and display commands on the same line. Therefore, most of the examples presented in this section combine ED commands.

When you combine several commands on a line, ED executes them in the same order they are entered, from left to right on the command line. There are four restrictions to combining ED commands:

- The combined command line may not exceed 128 characters.
- If the combined command line contains a character string, the string may not exceed 100 characters.
- When a command with a character string argument is to be followed by another command, the character string must be terminated by a ^Z to separate it from the next command letter. if the string is the last item on the command line, it may be terminated with a <cr>.
- Commands to terminate an editing session (E, H, 0, Q) may not appear in a combined command line.

Combined command may be grouped together in a macro and repeated a specified number of times. Details of combining ED commands and using macros are given in Section 12.4.8.

You may use the MP/M II control character functions to edit commands entered to the ED * prompt. The control characters ate defined in Section 21 an,..' are summarized in Appendix C. if you are inserting text into tint. file, the editing control characters work normally, but ^P (echo console output at printer) , ^S (pause console scroll) and ^C (program abort) do not work.

The following table alphabetically summarizes ED commands and their valid arguments. The table also indicates in parentheses which section describes each command in detail.

**Table 12-2. ED Command Summary**

| Command | Action |
|---|---|
| NA | Append n lines from original file to memory buffer (12.4.2) |
| B,- B | move CP to the beginning (B) or bottom (B) of buffer (12.4.3) |
| NC, -nC | move CP n characters forward (C) or back (C) through buffer (12.4.3) |
| nD,- nD | Delete n characters before (D) or from (D) the CP (12.4.5) |
| E | save new file and return to MP/M 11 (12.4.9) |
| **Fstring^z** | find character string (12.4.6) |
| H | save the new file, then reedit, using the new file as the original file (12.4.9) |
| I<cr> | enter insert mode (12.4.2) |
| **Istring^Z** | insert string at CP (12.4.2) |
| **Jsearch_string^Zinsert_string^Zdelete_to_string** | juxtapose strings (12.4.6) |
| nK, -nK | delete (kill) n lines from the CP (12.4.5) |
| NL, -nL, 0L | move CP n lines (12.4.3) |
| **NMcommand** | execute commands n times (12.4.8) |
| n, -n | move CP n lines and display that line (12.4.4) |
| n: | move to line n (12.4.3) |
| :ncommand | execute command through line n (12.4.3) |
| Nstring | extended find string (12.4.6) |
| 0 | return to original file (12.4.9) |
| nP, -nP | move CP 23 lines forward and display 23 lines at console (12.4.4) |
| Q | abandon new file, return to MP/M 11 (12.4.9) |
| R | read $$$.LIB file into buffer (12.4.7) |
| Rfilename | read filename.LIB into buffer (12.4.7) |
| **Sdelete_string^Zinsert_string^z** | substitute string (12.4.6) |

**Table 12-2 (continued)**

| Command | Action |
|---|---|
| nT, -nT, 0T | type n 'lines (12.4.4) |
| U, -U | uppercase translation (12.4.2) |
| V, V, 0V | line numbering on/off (12.4.1), display free bufferspace (12.4.2) |
| nW | write n lines to new file (12.4.9) |
| nX | write n lines to temporary LIB file (12.4.7) |
| nZ | wait n seconds (12.4.6) |

### 12.4.1 Line Numbers in the Memory Buffer

To help you keep track of the data in the memory buffer, ED call number lines of text when it displays data at the console. ED comes up with line numbering turned on, as shown below:

```
A>
 ED B:LETTER.TXT
       *#a
```

The line number before the * prompt indicates which line currently contains the CP, although the CP may be pointing to any character within the line.

Line numbers exist. only in the buffer and never become a part of a disk file. As you add or delete lines, ED dynamically renumbers the lines in the file. For example, if you add two lines to the middle of the data buffer, ED automatically adds two to the line numbers of all lines following the insert. Line numbering and renumbering occurs in EE internally whether the line number display is enabled or disabled.

To turn the line number display on or off, use the V command. The formats of the V command are:

```
V, V
```

Use a V command to disable the line number display. You call turn the line number display on or off at any time during an editing session.

Remember that ED defines a line as any characters after c <cr><lf> sequence up to and including the next <cr><lf>. It is most convenient to break up your text file into lines less than or equal to the line length of your console. If you enter lines longer than your console width, ED still assigns a line number only after a

<cr><lf> sequence, even if the display wraps around to the next line at the console. Note that the line concept is optional in ED; you may enter characters in an infinite string. However, lines and line numbers make it easier to keep track of the CP's location in the memory buffer.

### 12.4.2 Inserting Text into the Memory Buffer

ED supports three commands that add text to the memory buffer: I (insert) , A (append) , and R (read) . I inserts new characters entered at the keyboard into the memory buffer at CP. A copies lines in from the original disk file into the memory buffer, appending them to any text already in the buffer. R reads an entire file of type LIB into the memory buffer at CP. A fourth command discussed in this section, U, translates lowercase letters to uppercase when ED is in insert mode.

Before you add text to the buffer, you may want to check the amount of free space available. To do this, precede the line numbering command with the special 0 character, as shown below:

```
    1: *OV
25000/30000
    1: *
```

In the example above, ED reports that the buffer size is 30,000 characters and currently there is space for 25,000 new characters. From ED's display You can calculate that there are already 5,000 characters in the buffer.

To insert new characters in the memory buffer, use an I command An I command may take one of two forms:

or

    Istring ^Z

When the first form is given, ED enters insert mode. In this mode, all keystrokes are added directly to the memory buffer. Characters are inserted just before the CP, so imagine that the CP moves down as new data is added. The, following is an example of insert mode:

```
A>ED LETTER.TXT B:

NEW F1 L E
    : * i
  1: Compudealer
  2: 123 W. Fourth St.
  3: Inglevale, CA
  4:
  5: Dear Compudealer:
  6 :
  7: 1 am enjoying the MP/M system I received from
```

```
 8:  you last week. I am learning to use the MP/M
 9:  utilities, but know that soon I'll want
10:  more programs. Please send me a list of
11:  the MP/M software you keep in stock.
12:  Thank you for your help.
13:
14:  Sincerely,
15:  ^Z
```

In the example above, the buffer was empty, so there was no question as to the location of the CP. Note however, that if you enter insert mode when the CP is between two characters in the middle of a line, the insert does not start on a new line. Insert mode always inserts characters just before the CP and does not start a new line until you press the carriage return key. Use ^Z to exit from insert mode. The return of the ED prompt, *, indicates that ED is out of the insert mode.

Remember that in insert mode you can use MP/M II line editing control characters to edit your input without returning to ED's command level. Use ^H for destructive backspace, and ^U or ^X to delete a line. A RUB keystroke also deletes the character to the left of the cursor, but echoes the character at the console. Note that ^H cannot delete a <cr><lf> sequence to return to the previous line, but RUB can. However, you must use two RUB keystrokes, one for the <cr> and one for the <lf>.

You may also Enter ^I to tab characters into columns. ED's tab stops are at columns 9, 17, 25, 33, 41, 49, 57, and so on for extra - wide terminals. You cannot change ED's tab stops.

The second form of the I command does riot enter insert mode. It inserts the character string into the memory buffer and returns immediately to the ED prompt. In the example below, assume the CP is at the beginning of line 11. The command 8c moves the CP from the beginning of the line to just after the last M in MP/M. The i command inserts the string "compatible"; Olt retypes the line. ^Z concludes the insert string.

```
11:  the MP/M software you keep in stock.
11: *8cicompatibleZOlt

11:  the MP/Mcompatible software you keep in stock.
```

You may conclude an insert string by pressing <cr> instead of ^Z, but ED inserts a <cr><lf> sequence at the current location of the CP before returning to the * prompt. For example:

```
    11:  the MP/M software you keep in stock.
    11: *8ci-compatible<cr>
*d12t
    11: the MP/M-compatible
    12: software you keep in stock.
```

```
11: *
```

When entering & combination of numbers and letters, you may find it inconvenient to press a caps lock key at, your terminal if your terminal translates caps locked numbers to special characters. ED provides two ways to translate your alphabetic input to uppercase without affecting numbers. The first is to enter the insert command letter in uppercase: I. All alphabetics entered during the course of the capitalized command, either in insert mode or as a string argument, are translated to uppercase. For example:

```
9: utilities, but know that soon I'll
9: *9cl pip, ed, and stat ^ZOlt

9:  utilities PIP, ED AND STAT, but know that soon I'll
```

The second method is to enter a U command before inserting text. Uppercase translation remains in effect until you enter a U command. U also translates text inserted by other ED commands such as A and R, which are discussed below. F, N, J and S are discussed in Section 12.4.6 and are also affected by a U command.

When you invoke ED and specify an existing file to edit, ED checks to see that the file exists and opens it for copying. However, ED does not copy the file into the memory buffer until you specify the number of lines to be copied with an A command. If ED successfully appends the number of lines you specify, the CP points to the first character appended. The general form for an A command is:

```
nA
```

where the optional n may be a positive integer or one of the special characters # or 0. If you do not specify n, ED appends one line from the original file to the memory buffer. When you specify a number, ED appends that number of lines to the memory buffer. When you specify #, ED reads the entire file into the buffer. If ED fills the memory buffer before it copies the whole file, it issues an error message because it was unable to complete the command. The error message is:

```
BREAK ">" 1, A
```

The > symbol indicates that the memory buffer overflowed during execution of the A command. Because the append was unsuccessful, the CP points to the end of the memory buffer. To recover from this situation, use the W command described in Section 12.4.9.

The special 0 character can prevent memory overflow and spare you from calculating an exact number of lines to copy into the buffer. In response to OA, ED appends lines to the buffer until the buffer is about half full. This leaves enough room for normal editing and inserts. However, if during the editing session you insert more text than the buffer can hold, you may receive an error message similar to the one below:

```
        BREAK ">" AT I
```

To recover from this error, use the W command described in Section 12.4.9.

The last way to add text to the memory buffer is to read in a library (.LIB) file with a R command. The format of the R command

```
        Rfilename
```

where filename is the name of a file with the filetype LIB. Do not enter the LIB filetype. No space is permitted between the command letter R and the filename. If you combine an R command with other commands, you must separate the filename from subsequent command letters with a ^Z. There are special considerations if R is to be used in a macro command; see Section 12.4.8 for an explanation.

R inserts the library file in front of the CP. Therefore, after the file is added to the memory buffer, the CP points to the same character it did before the read, although the character may be on a new line number.

You may create L T B files in individual ED sessions, or you may use the X command to create a LIB file without leaving or restarting ED. This is useful for moving text blocks as described in Section 12.4.7.

### 12.4.3 Displaying Buffer Contents at the Console

ED does not display the contents of the memory buffer at the console until you specify which part of the text you want to see. You must indicate which lines to display in relation to the CP' s current location. The T command displays text at the console without moving the CP; the P and n commands relocate the CP as they display lines at the console.

The T command types a specified number of lines from the CP a t the console. The formats of the T command are:

```
        nT, nT
```

where n specifies the number of lines to be displayed. If a negative number is entered, ED displays n lines before the CP. A positive number displays n lines from the CP. if no number is specified, ED types from the CP thru the end of the line. The CP remains in its original position no matter how many lines are typed.

If the CP is between two characters in the middle of the line, a T command types only the characters between the CP and the end of the line. Use this command to verify the CP's location. For example, when the 16C command moves the CP just to the left of the P in Please, a T command can verify that fact in two ways:

```
    10: more programs. Please send me a list of
    10 : *16ct
 Please send me a list of
    10: *-t
     9: utilities, but know that soon I'll want
    10: more programs. *
```

To fill your console screen with text display, specify nT with n equal to the number of lines on your console minus one for the returned * prompt. On a 24 line console for example, enter 23T. If you specify n greater than your console length, you can enter a. TS to stop the display, then a ^Q to continue scrolling.

Although you can display any amount of text .t the console with a T command, it is sometimes more convenient to "page" through the buffer, viewing whole screens of data and moving tire CP to the top of each new screen at the same time. To do this, use ED's P command. When executing a P command, ED assumes your console screen is 24 lines long and that your text lines are less than or equal to your screen's line length. The P command takes the following forms:

```
np, -nP
```

where the optional n is the number of pages to be displayed. If you do not specify n, ED moves the CP forward 23 lines arid then types the 23 lines following the CP. This leaves the CP pointing to the first character on he screen.

To display the current page without moving the CP, enter OP. The special character 0 prevents the movement of the CP. If you specify a negative number for n, P pages backwards towards the top of the file. For example, when you enter 1) ED displays the 23 lines ahead of the CP and moves the CP back 23 lines.

The n command is a simple abbreviation that both moves tire CP and displays the destination line. The formats of the n command are:

```
n, n
```

where n is the number of lines the CP is to be moved. In response to this command, ED moves tire CP forward or back the number of lines, specified, then prints only the destination line. A further abbreviation of this command is to enter no number at all. In response to a carriage return without ca preceding command, E D assumes an n command of 1, moves the CP down to the next line and prints it. Also, a  by itself moves the CP up one line.

### 12.4.4 Moving the Character Pointer (CP)

This section describes the B, C, L, n: and :n commands, which move the CP in useful increments but do not display the destination line. To verify that the CP has been, moved correctly, add one of the display commands described in Section 12.4.3.

The B command moves the CP to the beginning or bottom of the memory buffer. The formats of the B command are:

```
B, -B
```

Unlike other ED commands, a minus sign in a *B* command does not move the CP towards the beginning of the memory buffer. B moves the CP to the end or bottom of the memory buffer ; B moves the CP to the beginning of the buffer. For example:

```
8: you last week. I am learning to use the MP/M
9: utilities, but know that soon I'll want
8: *bt
1: Compudealer
1: *bt
 : *
```

As demonstrated in previous sections, the C command moves the CP forward or back the specified number of characters. The formats of the C command are:

```
nC, -nC
```

where n is the number of characters the CP 's to be moved. A positive number moves the CP towards the end of the line and the bottom of the buffer. A negative number moves the CP towards the top of the buffer. Combine a. C command with a T command to verity the CP's new location:

```
   10: more programs. Please send me a list of
   10: *16ct
Please send me a list of
   10:*-t
    9 : uti1ies , but know that SOOn I'll want
    8: *bt
    1:  Compdealer
    1: *-bt
    1: *
```

If you wish, you can enter an n large enough to move, the CP to a different line. Remember however , that each line is separated by the invisible characters <cr><lf>, and you must compensate for their presence.

The L command moves the CP the specified number of lines. After an L command, the ~'P always points to the beginning of a line. The formats of the L command are:

```
nL, -nL
```

where n is the number of lines the CP is to be moved. A positive number moves the CP towards the end of the buffer. A negative number moves the CP towards the beginning of the buffer. For example, the command -L moves the CP to the beginning of the previous line, even if the CP originally points to a character in the middle of the line. Use the special character 0 to move the CP to the beginning of the current line. For example:

```
    1:  *71t
    8:  you last week. I am learning to use the MP/M
    8:  *31t
   11:  the MP/M software you keep in stock.
   11: *8ct
software you keep in stock.
   11:  *0lt
   11:  the MP/M software you keep in stock.
   11: *-21t
    9:  utilities, but know that soon I'll want
    9:  *41t
   13:
   13:  *-1t
   12:  Thank you for your help.
  12:   *
```

To move the CP to the end of a line without calculating the number of characters in the line, combine an L command with a C command as shown below:

```
8: you last week. I am learning to use the MP/M
8: *1-6c4distandard^Z0lt

8: you last week. I am learning to use the standard
8: *
```

ED accepts a line number as a command to specify a destination for the CP. The format for the line number command is:

```
n :
```

where n is the number of the destination line. This command places the CP at the beginning of the specified line, and can be most useful when combined with other commands. However, remember when moving to a specific line number that ED dynamically renumbers text lines in the buffer each time a line is added or deleted. The number of the destination line you have in mind may change during editing.

The inverse of nhe line number command specifies that a command should be executed through a certain line. This command must be used with another zommand and has the following format:

```
:ncommand
```

where n is the line number through which the command is to be executed. The :n portion of this command does not move the CP, but the command that follows in may. For example, :nT does not move the CP, but :nL does. You can combine n: with :n to specify a range of lines through which a command should be executed. For example:

```
14: *5::12t
  5: Dear Compudealer:
  6:
  7: 1 am enjoying the MP/M system I received from
   8: you last week. I am learning to use the standard
 9: utilities PIP, ED, AND STAT, but know that soon I'll want
 10: more programs. Please send me a list of
 11: the MP/M compatible software you keep in stock.
 12: Thank you for your help.
15: *
```

### 12.4.5 Deleting Characters

To erase or delete characters from the memory buffer, you may choose between two ED commands. The K command deletes whole lines from the buffer. The D command deletes a specified number of characters and has the form:

```
nD, -nD
```

where n is the number of characters to be deleted. If no number is specified, ED deletes the character to the right of the CP. A positive number deletes multiple characters to the right of the CP, towards the bottom of the file . A negative number deletes characters to the left of the CP, towards the top of the file. For example:

```
9:  utilities, but know that soon I'll want
9: *1-2c-4dineed^ZOlt

9: utilities, but know that soon I'll need
9: *
```

You can also use a D command to delete the <cr><lf> between two lines, joining them together as shown below:

```
1: *31t
1:  Compudealer
2:  123 W. Fourth St.
3:  Inglevale, CA
1: *1-2di ^Zl2di ^Z01t
1:  Compudealer 123 W. Fourth St. Inglevale, CA
1:  *
```

The K command deletes or "kills" data lines and takes the form:

```
nK, -nK
```

where n is the number of lines to be deleted. A positive number kills lines after the CP; a negative number kills lines before the CP. When no number is specified, ED kills the current line after the CP. If the CP is in the middle. of a line, a K command kills only the characters from the CP to the end of the line and concatenates the characters before the CP with the next line. A -K command deletes all the characters between the beginning of the previous line and the CP.

You may use the special # character to delete all the data from the CP to the beginning or end of the buffer. Or, to delete a block of text, use n::nK. Before using any K command, make sure any lines you may want to save are either written out to the new file or saved in a backup file. You cannot reclaim lines after they are removed from the buffer.

Remember that after a K command is executed, all the lines following the CP are renumbered. Take this into account if using a specific line reference in a subsequent ED command.

### 12.4.6 Finding and Replacing Strings

Because ED renumbers the data lines frequently as you edit, you may at some time find yourself knowing the word or phrase you want to edit next but unsure of its location in the buffer. To help you find it, ED supports a "find" command, F, that searches through tile buffer for you and places the CP after the phrase you want. The N command expands on this function, allowing ED to search through the entire source file instead of just the buffer. The F command performs the simplest find function. Its form is:

```
nFstring
```

where n is the occurrence of the string to be found. Any number you enter must be positive because ED can only search from the CP to the bottom of the buffer. If you enter no number, ED finds the next occurrence of the string in the file. Note that if you follow an F, command with another ED command on the same command line, you must terminate the string with a ^Z. In the following example, F finds the second occurrence of "I":

```
     9: *#t
     1:  Compudealer
     2:  123 W. Fourth St.
     3:  Inglevale, CA
     4:
     5:  Dear Compudealer:
     6:
     7:  1 an enjoying the MP/M, system I received from
     8:  yo,; last Week. I am learning to use the MP/M
     9:  utilities, but know that soon I'll want
    10:  more programs. Please send me a list of
    11:  the MP/M-compatible software you keep in stock.
    12:  Thank you for your help.
    13:
    14:  Sincerely,
     1:  *2fI^Zt
am enjoying the MP/M system I received from
    7: *
```

It makes a difference whether you enter the F command in upper or lower case. If you enter F, ED internally translates the argument string to upper case and finds the string only if it is in uppercase. If you specify f, ED looks for an exact match. For example, FMp/m searches for MP/M but fMp/m searches for Mp/m and will not find MP/M or mp/m. Note that if uppercase translation has been enabled by a U command, ED searches for an uppercase string whether the find command is specified F or f. If ED does not find a match for the argument string in the memory buffer, it issues the message:

**BREAK "#" AT (LINE #)**

where the symbol # indicates that the search failed during the execution of an F command.

The N command extends the search function beyond the memory buffer to include the original file. if the search is successful, it leaves the CP pointing to the first character after the search string. The form of the N command is:

```
nNstring
```

where n is the occurrence of the string to be found. if no number is entered, ED looks for the next occurrence of the string in the file. The case of the N command letter and whether or not uppercase translation is enabled by the U command have the same effect on an N command as they do _~n an F command . Note that if you follow an N command with another ED command, you must terminate the string with a ^Z.

When an N command is executed, ED searches the memory buffer for the specified string, but if ED doesn't find the string, it doesn't issue an error message. Instead, ED automatically writes the searched data from the buffer into the new file. Then it performs an OA command to fill the buffer with fresh data from the original file. ED continues to search the buffer, write out data and append fresh data until it either finds the string or reaches the end of the source file. if it reaches the end of the source file, ED issues the following message:

```
BREAK # AT N
```

Because ED writes the searched data to the new file before looking for more data in the original file, ED usually writes the contents of the buffer to the new file before finding the end of the original file and issuing the error message. An editing session cannot continue after the original file is exhausted and the memory buffer is emptied, so you must use the H command described in Section 12.4.9 to restart edit.

The S command searches only the buffer for the specified string, but when it finds it, automatically new string for the search string. This simplifies "global" changes, such as changing all occurrences of a customer name in a letter, a project name in a manual, or a label in a program. The form of the s command is:

```
nSsearch strinq^Znew string
```

where n is the number of substitutions to make. If no number is specified, ED searches for the next occurrence of the search string in the memory buffer. If uppercase translation is enabled by a U command or by a capital S command letter, ED looks for a capitalized search string and inserts a capitalized insert string. Note that if you combine this command with other commands, you must terminate the new string with a ^Z. in the following example, S substitutes the new string "Digital Research" for the search string "MP/M.''

```
 1: *6::1lt
 6:
 7:  I am enjoying the MP/M system I received from
 8:  you last week. I am learning to use the MP/M
 9:  utilities, but know that soon I'll want
10: more programs. Please send me a list of
11: the MP/M-compatible software you keep in stock.
 6: *sMP/M^ZDigital Research^Z6:11t
 6:
 7:  1 an enjoying the Digital Research system I
     received from
 8:  you last week. I am learning to use the MP/M
 9:  utilities, but know that soon I'll want
10: more programs. Please send me a list of
11: the MP/M-compatible software you keep in stock.
 6: *
```

The J command inserts a string after the search string, then deletes any characters between the end of the inserted string to the beginning of a third "delete to" string. This replaces the string between the search and delete to strings with the insert string. The form of the J command is:

```
nJsearch string^Zinsert strinq^Zdelete to string
```

where n is the occurrence of the search string. If no number is specified, ED searches for the next occurrence of the search string in the memory buffer. if uppercase translation is enabled by either a U command Pr an uppercase J command letter, ED looks for upper case search and delete to strings and inserts an uppercase insert string. Note that if you combine this command with other commands, you must terminate the delete to string with a ^Z.

The delete to string is optional in a J command, but if you don't use it, you must terminate the command with two ^Z's. In the example below, a J command replaces Digital Research for MP/M.

```
7: 1 am enjoying the MP/M system I received from
7: *jthe ^ZDigital Research ^Zsystem^ZOlt
7: 1 am enjoying the Digital Research system I
received from
7: *
```

The J command is especially helpful when revising comments in assembly language source code. Use a semicolon for the search string, a TL to represent <cr><lf> as the delete to string, and the new comment as the insert string, as shown below.

```
236:  SORT    LXI     H, SW    ;ADDRESS TOGGLE SWITCH
236:  *j;^ZADDRESS SWITCH TOGGLE^Z^L^ZOlt
236:  SORT    LXI     F1, SW   ;ADDRESS SWITCH TOGGLE
236: *
```

In any search string for a F, N, S or J command, you may use T to represent a <cr><lf> when your desired phrase extends across a line break. You may also use TI in a search string to represent a tab. At the console, the cursor moves to the next tab stop.

If multiple tabs or long strings make your command line longer than your console line length, you may enter a ^E to cause a physical carriage return at the console. A ^E returns the cursor to the left edge of tile console, but. does not send the command line to ED. When you finish your command, press the carriage return key to send the command to ED. Remember that no ED command line containing strings may exceed 100 characters.

### 12.4.7 ED Macro Commands

Combined ED commands are powerful, but an ED macro command, M, can increase the usefulness of a string of commands by executing it a specified number of times. For example, 3FMP/M^ZOLT types the line that contains the third occurrence of MP/M from the CP, but 3MFMP/M^ZOTT displays the lines containing all three occurrences. The form of the M command is:

```
nMcommand string
```

where n is the number of times the command string is to be executed. A negative number is not a valid argument for an M command. If no number is specified, the special # character is assumed and ED executes the command string until it reaches the end of data in the buffer or the end of t:.. original file, depending on the commands specified in the string.

The terminator for an M command is a carriage return; therefore, an M command must be the last command on the line. Also, all character strings that appear in a macro must be terminated by ^Z . If a character string ends the combined command string, it must be terminated by ^Z, then followed by a <cr> to end the M command. in the following example. a macro repeats combined substitute and display commands.

```
 1 : *7::1lt
 7:  1 am enjoying the Digital Research system
     I received from
 8:  you last week. I am learning to use the Digital
     Research
 9:  utilities, but know that soon I'll want
10: more programs. Please send me a list of
11: the Digital Research software you keep in stock.
 7: *3msDigital Research^ZMP/M^ZOlt
 7:  1 am enjoying the MP/M system I received from
 8:  you last week. I am learning to use MP/M
11: the MP/M software you keep in stock.

BREAK "#" AT T
   11: *
```

The execution of a macro command always ends in a BREAK message, even when you have limited the number of times the macro is to be performed and ED does not reach the end of the buffer or original file. Usually the command letter displayed in the message is one of the commands from the string and not M.

Certain command sequences at the end of the command string guarantee that changes are made the way you want them. For example, the sequence OLTL at the end of a substitute macro insures that only one substitution is made per line. For example:

```
 1:  Compudealer
 2:  123 W. Fourth St.
 3:  Inglevale, CA
 4:
 5:  Dear Compudealer
 6:
 7:  1 am enjoying the Digital Research system I
     received from
 8:  you last week. I am learninng to use the MP/M
 9:  utilities, but know that soon I'll want
10: more programs. Please send me a list of
11: the MP/M-compatible software you keep in stock.
12: Thank you for your help.
13:
14: Sincerely,
 1: *7:3ms.^Z;^ZOltl
 8:  you last week; I am learning to use the MP/M
10: more programs; Please send me a list of
11: the MP/M-compatible software you keep in stock;

BREAK "#" AT T
   11: *
```

when the sequence OT concludes a substitute macro, it insures that every occurrence of the search string is replaced, even if there are several occurrences on the same line.

To abort a macro command, strike any key at the console . However, even if you enter display commands in the command string, a macro usually executes too quickly for you to abort the command in anything but a haphazard fashion. To make the abort facility more useful, ED supports a~ "wait" command, Z, in macros. The format of the Z command is:

```
nZ
```

where n is the number of seconds ED waits before executing the next command in the string. The number must be positive. A combination of F, Z and T commands can give you time to read a display of each occurrence of the search string. When you find the one you want to edit, press any key to abort the macro. In the following example, the user has 5 seconds to abort when the string he wants is displayed. He aborts the macro by pressing the y key.

```
       1: *mfMP/M^ZOt5z
       6:
       7:  1 am enjoying the MP/M
       7:  1 am enjoying the MP/M system I received from
      8:  you last week. I am learning to use the MP/M
     10:  more programs. Please send me a list of
     11:  the MP/My
BREAK"#" AT ^z
          11: *
```

To verify a search and replace command, you must combine four commands in a macro: F or N to find the string, -nT to type enough of the context for you to make a decision, nZ to make ED wait while you decide, and I to insert the change if you decide not to abort. This has the advantage of letting you specify N for the search command, which carries the macro through the remainder of the original file instead of limiting your command to the contents of the memory buffer. Of course, if ED finds and displays an occurrence you do riot ,,,ant to change and you abort the macro, you have to reenter the macro command line to continue substituting.

Be extra careful when including an R command in an unlimited macro. If ED reaches the end of data in the data buffer and cannot execute any of the other commands combined in the macro string, it fills the remainder of the buffer by repeatedly inserting the LIB file. You can avoid this problem by combining a Z command with any R command that may appear in a macro and aborting the command when it reaches the end of data in the buffer.

### 12.4.8 Moving Text Blocks

To move a group of lines from one area of your data to another, use an X command to write the text block into a temporary .LIB file, then a K command to remove the lines from their original location, and then an R command t.) read the block into its new location. The format of the X command, is:

```
nX
```

where n is the number of lines from the CP towards the bottom of the buffer that are to be written into the temporary file. Therefore, n must always be a positive number. if no temporary file exists, ED creates one named X$$$$$$$.LIB. If X$$$$$$$.LIB exists when an X command is executed, ED appends the specified lines to the end of the existing file. Use the special character 0 as the n argument in an X command to delete an existing X$$$$$$$.LIB. ED creates a new temporary file with the same name the next time an X command is executed.

To read the X$$$$$$$.LIB file into its new location, first position the CP where you want the block to be inserted. Then enter an R command to read the temporary file. You do not need to enter a filename specification; when R is entered without a filename, ED automatically searches for X$$$$$$$.LIB. However, unless the R command is the last on the line, it must be followed by a ^Z so that ED does not try to interpret subsequent commands as a filename.

In the following example, the user must edit the sentence he wants to move so that it starts and ends on line boundaries. This procedure can add extra editing time if you are moving sentences, but if you are editing line oriented text such as a program, extra editing is unnecessary.

```
 8: *you last week. I am learning to use the MP/M
 8: *16ci
 9:
 9: *9::Ilt
 9: 1 am learning to use the MP/M
10: utilities, but know that soon I'll want
11: more programs. Please send me a list of
 9: *2116ci
11:
12:
12:  *9::13t
 9: 1 am learning to use the MP/M
10: utilities, but know that soon I will want
11: more programs.
12: Please send me a list of
13: the MP/M software you keep in stock.
 9: *3x3kb#t
 1: Compudealer
 2: 123 W. Fourth St.
 3: Inglevale, CA
 4:
 5: Dear Compudealer:
 6:
 7:1 am enjoying the MP/M system I received from
 8: you last week.
 9: Please send me a list of
10:  the MP/M software you keep in stock.
11: Thank you for your help.
12:
```

```
13:  Sincerely,
 1:  *11:r^Z7::14t
 7:  1 am enjoying the MP/M system I received from
 8:  you last week.
 9:  Please send me a list of
10:  the MP/M software you keep in stock.
11:  1 am learning to use the MP/M
12:  utilities, but know that soon I'll want
13:  more programs.
14:  Thank you for your help.
7:   *
```

ED normally deletes temporary files at the end of an editing session. For example, an exit performed by an E or Q command erases any file named X$$$$$$$.LIB. If you exit ED by a ^C, the temporary file remains on the disk. However, if you want to preserve that file, you must rename it before the next editing session. When invoked, ED erases any file named X$$$$$$$.LIB.

### 12.4.9 Saving or Abandoning Changes: ED Exit

In general, you can to save or abandon editing changes while the data is still in the buffer. You can save your editing changes by instructing ED to write the contents of the data buffer to the new file. The W and H commands perform this task without ending the ED session; an E command saves the contents of the buffer and exits ED.

You can abandon your editing changes with an 0 command, which moves the MP back to the top of the buffer without writing out its contents, or with a Q command, which exits ED without saving the changes in the new file. You may also end an ED session by a ^C (program abort) to return directly to MP/M II.

Except for the W command, none of these commands requires an argument. Also, any command that terminates an ED session must be the only command on the line.

The V~ command is similar to the A command in that it transfers a specified number of lines at a time. However, instead of transferring lines from the original file to the buffer, the W command writes lines from the buffer to the new file. The form of the W command is:

```
nW
```

where n is the number of lines to be written from the top of the buffer to the bottom of the new file. This must. be a positive number, or the special character 0 to write approximately half the contents of the data buffer to the new file. ED inserts the lines just before the ^P in the new file.

Use the W command to make room in the buffer for more lines from the original file. After a W command is executed, you can no longer edit the saved lines.

An H command also saves the contents of the data buffer without ending the ED session, but its purpose is to return to the "head" of the file. It saves the current changes and lets you reedit the file without exiting ED.

To execute an H command, ED first finalizes the new file, transferring all lines remaining in the buffer and the original file to the new file . Then ED closes the new file, erases any BAK file that has the same filename as the original file, and renames the original file filename.BAK. ED then renames the new file, which has had the extension .$$$, with the original filename. Finally, ED opens the newly renamed file as the original file for a new edit, and opens a new .$$$ file. When ED returns the * prompt, the CP is at the beginning of an empty buffer. In short, an H command has the same effect as an E command followed by a second invocation of ED with the same filename.

An E command performs a normal exit from ED. To execute an E command, ED first writes all data lines from the buffer and the original file to the new file. If a BAK file exists, ED deletes it, then renames the original file to BAK. Finally, ED renames the new file from .$$$ to the original filename extension and returns control to the CCP.

The operation of the E command makes it unwise to edit a file with the extension BAK. When you edit a BAK file and exit with an E command , ED erases your original file because it has a BAK extension. Then, when ED can't find the original file to rename with the BAK extension, it aborts without saving the new file. To avoid this, always rename a BAK file with some other extension before editing it with ED.

To purposely abandon a new file, use an 0 or Q command. An 0 command abandons changes made since the beginning of the edit and allows you to reedit without ending the ED session. When you enter an 0 command, ED confirms that you want to abandon your changes by asking:

```
0- (Y/N) ?
```

When you enter Y, ED erases the .$$$ file and the contents of the memory buffer, moves the SP back to the beginning of the original file, and opens a new .$$$ file. When the * prompt returns, the CP is pointing to the beginning of an empty memory buffer, just as it is when you invoke ED.

A Q command abandons changes made since the beginning of the ED session and exits ED. When you enter a Q command, ED verifies that you want to abandon the changes by asking:

```
Q- (Y/N) ?
```

When you enter Y, ED erases the .$$$ file, closes the original file and returns control to the CCP. The original file is not renamed or changed in any way.

You may also enter a ^C to return control to MP/M II. This does not give ED a chance to close the original or new files, but it prevents ED from deleting any temporary files.

## 12.5 ED Error Messages

ED may return one of two types of error messages: an ED message if ED 'cannot execute an edit command, or a MP/M II message if ED cannot read or write to the specified file. The format for an ED error message is:

```
BREAK "x" AT c
```

where x is one of the symbols defined in Table 12-3 and c is the command letter where the error occurred.

**Table 12-3. ED Error Symbols**

| Symbol | Meaning |
|---|---|
| # | Search failure. ED cannot find the string specified in a F, S, or N command. |
| ? | Unrecognized command letter c. ED does not recognize the indicated command letter, or an E, H, Q or 0 command is not alone on its command line. |
| 0 | No LIB file. ED did not find the LIB file specified in an R command. |
| > | Buffer full. ED cannot put any more characters in the memory buffer, or string specified in an F, N, or S command is too long. |
| E | Command aborted. A keystroke at the console aborted command execution. |

The following examples show how to recover from common editing error conditions. For example:

```
BREAK ">" AT A
```

means that ED filled the buffer before completing the execution of an A command. When this occurs, the CP is at the end of the buffer.

Use the W command to write lines from the beginning of the buffer to the temporary file.

```
BREAK "#" at F
```

means that ED reached the end of the memory buffer without matching the string in an F command. At this point, the CP has not been moved from its location when the F command was made.

Table 12-4 below defines the disk file error messages ED returns when it cannot read or write a file.

**Table 12-4. ED Disk File Error Messages**

| Message | Meaning |
|---|---|
| Bdos Error On d: R/O | Disk d: has read-only attribute. This occurs if a different disk has been inserted in the drive since the last cold or warm boot; message also occurs if the drive has been set to Read Only status by STAT. |
| DISK OR DIRECTORY FULL<br>FILE ERROR | Disk or directory full, ED cannot write anything more on the disk. This is a fatal error, so make sure there is enough space on the disk to hold a second copy of the tile before invoking ED. |
| FILE IS READ ONLY | The file specified in the command to invoke ED has the Read Only attribute. ED can read the file so that the user can examine it, but ED cannot change an Read Only file. |

I

# Appendix A
# ASCII and Hexadecimal Conversions

ASCII stands for American Standard Code for Information Interchange. The code contains 96 printing and 32 non-printing characters used to store data on a disk. Table A-1 defines ASCII symbols, then Table A-2 lists the ASCII and hexadecimal conversions. The table includes binary, decimal, hexadecimal, and ASCII conversions.

**Table A-1. ASCII Symbols**

| Symbol] | Meaning | Symbol | Meaning |
|---|---|---|---|
| ACK | acknowledge | FS | file separator |
| BEL | bell | GS | group separator |
| BS | backspace | HT | horizontal tabulation |
| CAN | cancel | LF | line feed |
| CR | carriage return | NAK | negative acknowledge |
| DC | device control | NUL | null |
| DEL | delete | RS | record separator |
| DLE | data link escape | SI | shift in |
| EM | end of medium | SO | shift out |
| ENQ | enquiry | SOH | start of heading |
| EOT | end of transmission | SP | space |
| ESC | escape | STX | start of text |
| ETB | end of transmission | SUB | substitute |
| ETX | end of text | SYN | synchronous idle |
| FF | form feed | US | unit separator |
| | | VT | vertical tabulation |

**Table A-2. ASCII Conversion Table**

| Binary | Decimal | Hexadecimal | ASCII |
|---|---|---|---|
| 0000000 | 0 | 0 | NUL |
| 0000001 | 1 | 1 | SOH (CTRL-A) |
| 0000010 | 2 | 2 | STX (CTRL-B) |
| 0000011 | 3 | 3 | ETX (CTRL-C) |
| 0000100 | 4 | 4 | EOT (CTRL-D) |
| 0000101 | 5 | 5 | ENQ (CTRL-E) |
| 0000110 | 6 | 6 | ACK (CTRL-F) |
| 0000111 | 7 | 7 | BEL (CTRL-G) |
| 0001000 | 8 | 8 | BS (CTRL-H) |
| 0001001 | 9 | 9 | HT (CTRL-1) |
| 0001010 | 10 | A | LF (CTRL-J) |
| 0001011 | 11 | B | VT (CTRL-K) |
| 0001100 | 12 | C | FF (CTRL-L) |
| 0001101 | 13 | D | CR (CTRL-M) |
| 0001110 | 14 | E | SO (CTRL-N) |
| 0001111 | 15 | F | SI (CTRL-0) |
| 0010000 | 16 | 10 | DLE (CTRL-P) |
| 0010001 | 17 | 11 | DC1 (CTRL-Q) |
| 0010010 | 18 | 12 | DC2 (CTRL-R) |
| 0010011 | 19 | 13 | DC3 (CTRL-S) |
| 0010100 | 20 | 14 | DC4 (CTRL-T) |
| 0010101 | 21 | 15 | NAK (CTRL-U) |
| 0010110 | 22 | 16 | SYN (CTRL-V) |
| 0010111 | 23 | 17 | ETB (CTRL-W) |
| 0011000 | 24 | 18 | CAN (CTRL-X) |
| 0011001 | 25 | 19 | EM (CTRL-Y) |
| 0011010 | 26 | 1A | SUB (CTRL-Z) |
| 0011011 | 27 | 1B | ESC (CTRL-[) |
| 0011100 | 28 | 1C | FS (CTRL-\) |
| 0011101 | 29 | 1D | GS (CTRL-]) |
| 0011110 | 30 | 1E | RS (CTRL-^) |
| 0011111 | 31 | 1F | US (CTRL--) |
| 0100000 | 32 | 20 | (SPACE) |
| 0100001 | 33 | 21 | ! |
| 0100010 | 34 | 22 | " |
| 0100011 | 35 | 23 | # |
| 0100100 | 36 | 24 | $ |
| 0100101 | 37 | 25 | % |
| 0100110 | 38 | 26 | & |
| 0100111 | 39 | 27 | ' |
| 0101000 | 40 | 28 | ( |
| 0101001 | 41 | 29 | ) |
| 0101010 | 42 | 2A | * |
| 0101011 | 43 | 2B | + |
| 0101100 | 44 | 2C | ` |
| 0101101 | 45 | 2D | - |
| 0101110 | 46 | 2E | . |
| 0101111 | 47 | 2F | / |
| 0110000 | 48 | 30 | 0 |
| 0110001 | 49 | 31 | 1 |
| 0110010 | 50 | 32 | 2 |

**Table A-2. (continued)**

| Binary | Decimal | Hexadecimal | ASCII |
|---|---|---|---|
| 0110011 | 51 | 33 | 3 |
| 0110100 | 52 | 34 | 4 |
| 0110101 | 53 | 35 | 5 |
| 0110110 | 54 | 36 | 6 |
| 0110111 | 55 | 37 | 7 |
| 0111000 | 56 | 38 | 8 |
| 0111001 | 57 | 39 | 9 |
| 0111010 | 58 | 3A | : |
| 0111011 | 59 | 3B | ; |
| 0111100 | 60 | 3C | < |
| 0111101 | 61 | 3D | = |
| 0111110 | 62 | 3E | > |
| 0111111 | 63 | 3F | ? |
| 1000000 | 64 | 40 | @ |
| 1000001 | 65 | 41 | A |
| 1000010 | 66 | 42 | B |
| 1000011 | 67 | 43 | C |
| 1000100 | 68 | 44 | D |
| 1000101 | 69 | 45 | E |
| 1000110 | 70 | 46 | F |
| 1000111 | 71 | 47 | G |
| 1001000 | 72 | 48 | H |
| 1001001 | 73 | 49 | I |
| 1001010 | 74 | 4A | j |
| 1001011 | 75 | 4B | K |
| 1001100 | 76 | 4C | L |
| 1001101 | 77 | 4D | M |
| 1001110 | 78 | 4E | N |
| 1001111 | 79 | 4F | 0 |
| 1010000 | 80 | 50 | P |
| 1010001 | 81 | 51 | Q |
| 1010010 | 82 | 52 | R |
| 1010011 | 83 | 53 | S |
| 1010100 | 84 | 54 | T |
| 1010101 | 65 | 55 | U |
| 1010110 | 86 | 56 | V |
| 1010111 | 87 | 57 | W |
| 1011000 | 88 | 58 | X |
| 1011001 | 89 | 59 | Y |
| 1011010 | 90 | 5A | Z |
| 1011011 | 91 | 5B | [ |
| 1011100 | 92 | 5C | \ |
| 1011101 | 93 | 5D | ] |
| 1011110 | 94 | 5E | ^ |
| 1011111 | 95 | 5F | < |
| 1100000 | 96 | 60 | ' |
| 1100001 | 97 | 61 | a |
| 1100010 | 98 | 62 | b |
| 1100011 | 99 | 63 | c |
| 1100100 | 100 | 64 | d |

**Table A-2. (continued)**

| Binary | Decimal | Hexadecimal | ASCII |
|---|---|---|---|
| 1100101 | 101 | 65 | e |
| 1100110 | 102 | 66 | f |
| 1100111 | 103 | 67 | 9 |
| 1101000 | 104 | 68 | h |
| 1101001 | 105 | 69 | i |
| 1101010 | 106 | 6A | j |
| 1101011 | 107 | 6B | k |
| 1101100 | 108 | 6C | 1 |
| 1101101 | 109 | 6D | m |
| 1101110 | 110 | 6E | n |
| 1101111 | ill | 6F | 0 |
| 1110000 | 112 | 70 | p |
| 1110001 | 113 | 71 | q |
| 1110010 | 114 | 72 | r |
| 1110011 | 115 | 73 | s |
| 1110100 | 116 | 74 | t |
| 1110101 | 117 | 75 | U |
| 1110110 | 118 | 76 | v |
| 1110111 | 119 | 77 | w |
| 1111000 | 120 | 78 | x |
| 1111001 | 121 | 79 | y |
| 1111010 | 122 | 7A | z |
| 1111011 | 123 | 7B | { |
| 1111100 | 124 | 7C | \| |
| 1111101 | 125 | 7D | } |
| 1111110 | 126 | 7E | ~ |
| 1111111 | 127 | 7F | DEL |

# Appendix B
# File Types

MP/M II identifies every file by a unique file specification, which consists of a drive specification, a filename, a filetype , and an optional password. The filetype is an optional three character ending separated from the filename by a period. The filetype generally indicates a special kind of file. The following table lists common filetypes and their meanings.

**Table B-1. File Types**

| **Filetype** | **Indication** |
|---|---|
| ASM | Assembly language source file; the MP/M II assembler, ASM, assembles or translates a file oftype ASM into machine language. |
| BAK | Back-up file created by a text editor; an editor renames the source file with this filetype indicate that the original file has been processed. The original file stays on the disk as the backup file, so you can refer to it. |
| BRS | Banked Resident System Process f le; some Resident System Processes also need a file of type BRS to execute properly. |
| COM | Command file that contains instructions in machine executable code. |
| HEX | Program file in hexadecimal format. |
| IRL | Indexed REL file produced by LIB. |
| LST | Printable file that can be sent to a list device. |
| PRL | Page Relocatable file; a file that does not require one fixed memory location, but can occupy various locations in memory. |
| PRN | Print file that can be listed at the console or sent to a peripheral device. |
| REL | Relocatable file produced by RMAC, which can be linked by LINK-80. |

**Table B-1. (continued)**

| Filetype | Indication |
|---|---|
| RSP | Resident System Process file; a program included in the MP/M II operating system at system generation. A Resident System Process will always be available even though it is not on a disk. An RSP does not require a memory bank and can therefore be executed even though all the banks are allocated. |
| SPR | System Page Relocatable file; system files required to generate MP/M II, such as BNKBDOS.SPR, BNKXIOS.SPR, XDOS.SPR, XIOS.SPR, and RESBDOS.SPR. |
| SUB | File type required for SUBMIT program containing one or more MP/M II commands. The SUBMIT program executes the commands in the submit file providing a batch mode for MP/M II. |
| TEX | Source file for the Digital Research text formatter, TEX-80. |
| TXT | Text file. |
| XRF | Cross-Reference file produced by XREF. |
| $$$ | Temporary file. |
| $n$ | Temporary file created by SUBMIT, (n is a console number). |

# Appendix C
# MP/M II Control Character Summary

**Table C-1. MP/M II Control Characters**

| Keystroke | Action |
|---|---|
| RUB | deletes character to the left of cursor; Echoes character deleted - cursor moves right. |
| DEL | same as RUB. |
| BACKSPACE | moves cursor back one space, erases previous character. |
| ^H | same as BACKSPACE. |
| ^U | cancels line, displays #, cursor moves down one line and awaits a new command. |
| ^X | deletes all characters in command line. |
| ^R | retypes current command line; useful after using RUB or DEL key. |
| ^E | forces a physical carriage return, but does not send command to MP/M II. |
| RETURN | carriage return. |
| ^M | same as carriage return. |
| ^J | line feed, terminates input at the console. |
| ^Z | string or field separator. |
| ^P | echoes all console activity at the printer; a second ^P ends printer echo. This only works if your system is connected to a printer. |
| ^S | stops console listing temporarily; ^Q resumes the listing. |
| ^Q | resumes console listing after ^S. |
| ^C | prompts to abort a program currently running at a given console. |
| ^D | detaches the currently executing program fromthe console at which the ^D is entered. |

# Appendix D
# MP/M II Error Messages

Error messages come from several different sources. MP/M II displays error messages when there are errors in calls to its Basic Disk operating System (BDOS). MP/M II also displays messages when there are errors in command lines. Each utility program supplied with MP/M II has its own set of error messages. You will also see error messages from any other applications programs that you might be running. Table D-1 displays the BDOS error messages. BDOS error messages are displayed in the following format.

```
Bdos Err on d: Message
Function NNN File: FILENAME.TYP
```

"d" indicates the drive involved; "message" indicates the appropriate error message; NNN indicates the number of the BDOS function involved, and FILENAME.TYP indicates the file involved. Table D-2 displays the MP/M II error messages resulting from errors in command lines.

**Table D-1. MP/M II Error Messages**

Message Meaning

File Opened in Read/only Mode

An attempt was made to open a file in locked or unlocked mode which has already been opened in Read only mode.

File Currently Open

A process is trying to access a file which is already being accessed in other than read only mode.

Close Checksum Error

A file cannot be closed properly because the present directory does not match the one that is logged into memory. This probably means the disk has not been reset, or a program error has occurred.

**Table D-1. (continued)**

| Message | Meaning |
|---|---|
| Password Error | A password is incorrect or missing. |
| File Already Exists | An attempt was made to create or rename a file when there is already a file of that name and type on the disk. |
| Illegal ? in FCB | A wildcard character is being used where wildcard filenames are not permitted. |
| Open File Limit Exceeded | An attempt was made to open one more file than the maximum number of open files per process that the system can accommodate. |
| No Room in System Lock List | An attempt was made to lock one more record than the maximum number of locked records per process that the system can accommodate. |
| Bad Sector | This error occurs when there is an actual hardware error on the disk. It may occur as a. result of trying to read a disk of one density in a drive which is set to a different density, or with an improperly inserted floppy disk. |
| Select | A nonexistent drive has been selected, or there is no disk in the selected drive.. |
| R/O | An at tempt was made to write to a file when the file or the whole drive had been set to Read Only. |

**Table D-2. MP/M II System Error Messages**

| Message | Meaning |
| --- | --- |
| Prg ld err | Program load error: there is not a large enough absolute transient program area based at zero to load a file of type COM. |
| Abs TPA not free | Absolute Transient Program Area not free: there is not an absolute Transient Program Area available based at location zero. |
| Reloc seg not free | Relocatable program - segment not free: there is not a large enough memory segment in which to run a file of type PRL. |
| Blank file type rqd | Blank file type required: A command keyword in the command line has been specified with a file Type. |
| Queue full | Queue full: the queue specified in the command keyword cannot hold anymore messages. |
| Bad entry | Bad entry: an improperly formatted command keyword has been entered. |

# Appendix E
# Checklist for Using Files

If the drive is set to a different density than the disk inserted in it, MP/M II returns a Bad Sector error. (See Appendix D, MP/M II Error Messages.)

If the file is set to Read Only, you can read the file but you cannot write to the file.

If the drive is set to Read Only, you cannot write to files on that drive. This occurs if you forget to use DSKRESET before changing a disk.

If you have typed a TS, your keyboard is locked until you type a ^Q to unlock it.

If you receive a "reloc seg not free" message, use a TD to reattach a process to the console so it can finish executing and free a memory segment. This situation could also occur if you accidentally typed a TD and didn't realize it.

Files with the DIR attribute can only be accessed if they are in the default user area on the default or specified drive.

Files with the SYS attribute can be accessed if they are in the default user area or user 0 of the default or specified drive, or in the default or user 0 area of the system drive.

If a drive is specified MP/M II only looks for a file in the default and zero user areas of the specified drive.

If the command line specifies a drive or a password and the command identifies a queue associated with Resident System Process, MP/M II will not find the command.

If the file is password protected, you might get a password error message.

Is the password protection mode set to READ, WRITE, DELETE or NONE? (See the SET command in Section 6).

- If the password protection mode is set to READ, then you need a password to read the file.

- If the password protection mode is set to WRITE, you can read the file without supplying the password, but you need the password to write to the file.

- If the password protection mode is set to DELETE, you can read or write to the file, but you need the password to erase it.

- If the mode is set to NONE, the password is erased: you no longer need it at all.

* Does the drive have a label? (See the SET command in Section 6).

  _ If the drive has a label and password protection is turned on for the drive, then you need a password to access any password protected files on that drive.

# Appendix F
# MP/M II Command Summary

MP/M II is distributed with over thirty utilities. Table F-1 lists them alphabetically and includes ten utilities that are covered in the Programmer's Guide or the System Guide. These ten utilities are listed here for convenience. The required parts of the command line are printed in boldface.

Table F-1. **Command Summary**

Syntax — Definition and Examples

**ABORT programname** n

Stops execution of program programname initiated from console n.

```
OA>ABORT PIP
2B>ABORT CRUN 2
```

**ASM filename.shp**

```
s = source file drive (A-P)
h = hex file drive     (A-P,Z)
p = print file drive   (A-P,X,Z)
```

Assembles assembly language commands into hexadecimal format. ASM requires a filename, the file type is assumed to be ASM and should not be specified. Instead a filetype can be specified to select special drives for the input and output files. The first letter in the file type indicates which drive contains filename.ASM. The second letter in the file type indicates which drive receives the assembled HEX file. The third letter in the file type indicates which drive receives the PRN file. The special drive codes Z and X, mean "no output desired" and "output to the console," respectively.

```
OA>ASM PROG
OA>ASM PROG.ABX
OA>ASM PROG.AZZ
```

**Table F-1**. **(continued)**

| Syntax | Definition and Examples |
|---|---|
| ATTACH programname | Attaches a detached program designated by programname to console from which it was invoked.<br>OA>ATTACH GENLEDGR<br>3C>ATTACH BASIC |
| CONSOLE | Returns the number of the console at which the command is entered.<br>OA>CONSOLE |
| DDT filespec | Debugs assembly language programs of file type COM, HEX or PRL.<br>OA>DDT<br>OA>DDT PROG.HEX<br>OA>DDT B:PROG.HEX<br>OA>DDT B:PROG.COM |
| DIR filespec [SYS,Gn], filespec | Displays a directory of files on a disk. Accepts ambiguous filenames to display a group of similarly named files. The [SYS] option displays system files also. The [Gn] option displays the directory of user number n.<br>OA>DIR<br>OA>DIR B:<br>OA>DIR B:DRAFT.TXT<br>OA>DIR B:*.TXT [SYS]<br>OA>DIR B:DRAFT.* |
| **DSKRESET** d:,d:,d: | Logs out all or specified drives except for the default drive. This will reinitialize the drives the next time they are accessed. Done before disk changes.<br>OA>DSKRESET<br>OA>DSKRESET A:,B:,C: |

**Table F-1. (continued)**

| Syntax | Definition and Examples |
|---|---|
| DUMP filespec | Displays contents of the file in hexadecimal format at the console.<br>OA>DUMP DRAFT.TXT |
| ED filespec | Creates or edits programs and data files. Does not accept ambiguous filenames.<br>OA>ED DRAFT.TXT<br>OA>ED B:DRAFT.TXT<br>OA>ED F:DOCUMENT.LAW;SECRET |
| ERA filespec [XFCB] | Erases a file or a group of files. Accepts ambiguous filenames. With [XFCB] it Erases only XFCB's of specified files.<br>OA>ERA DRAFT.BAK<br>OA>ERA B:*.BAK<br>OA>ERA B:DRAFT.*;SECRET<br>OA>ERA B:*.*<br>OA>ERA B:DRAFT.* [XFCB] |
| ERAQ filespec [XFCB] | Same as ERA except it queries for each specified file before erasing.<br>OA>ERAQ F:*.*[XFCB]<br>3F>ERAQ *.COM;PASSWORD<br>2C>ERAQ D:DRAFT.*;SECRET |
| GENHEX filename.COM offset | Generates a file of type HEX from a file of type COM (the COM must be specified). The offset is the starting address of the HEX file.<br>1A>GENHEX TESTRUN.COM 100<br>2E>GENHEX TESTRUN2.COM;PASSWORD 100 |

**Table F-1. (continued)**

```
Syntax                  Definition and Examples
```

**GENNOD filename.HEX filenetme.PRL** $nnnn

```
                        Produces a PRL or RSP relocatable program from
                        two concatenated HEX files offset by 100H. The
                        file types of the HEX and PRL files must be
                        specified. $nnnn is optional and specifies the
                        amount of extra memory that GENMOD allots.
                             CA>GENMOD PROGRAM.HEX PROGRAM2.PRL
                             3F>GENMOD PROGRAM.HEX PROGRAM2.PRL $1000
```

**GENSYS** $A

```
                        Generates the file MPM.SYS to hold the MP/M II
                        system. GENSYS generates new versions fo the
                        system. Can be executed from CP/M or MP/M. If
                        the $A option is specified GENSYS goes through
                        an automatic system generation using the
                        defaults found in the SYSTEM.DAT file on the
                        default drive.
                             OA>GENSYS
                             OA>GENSYS $A
```

**LIB d:filename=d:filenamel,d:filename2 ...**
**LIB d:filename[optionl=d:filenaniel,d:filename2 ...**

```
option = I, M or P
                        This command creates a library of type REL or
                        IRL from a number of relocatable REL files
                        (produced by RMAC). The I option creates an
                        indexed library (IRL), the M option displays
                        the module names and the P option displays the
                        module names and publics.
                             OA>LIB ASMLIB=MODULEl,MODULE2,MODULE3
                             OA>LIB ASMLIB[M]

LINK d:filenamel, d:filename2, d:filename3 ...
                        LINK combines relocatable object modules into a
                        COM or PRL file ready for execution under MP/M
                        II. The [OS] option makes SPR files, such as
                        the BNKXIOS file used in generating an MP/M II
                        system.
                             OA>LINK BNKXIOS=MOD1,MOD2,MOD3,MOI)4L'OS]
```

**Table F-1. (continued)**

| Syntax | Definition and Example |
|---|---|
| **LOAD d:filename.HEX** | Converts hexadecimal format program file into an executable command file. Load expects a file of type HEX.<br>OA>LOAD FILE.HEX<br>OA>LOAD B:FILE.HEX |
| MPMLDR | Loads the MP/M II system f ile MPM.SYS into memory, relocates it, and executes MPM.SYS to bring up MP/M II.<br>OA>MPMLDR |
| MPMSTAT | Displays the internal status of MP/M II.<br>OA>MPMSTAT |
| PIP destination filespec [Gnl=source filespec [options] | Transfers information between peripheral devices and concatenates files.<br>OA>PIP<br>OA>PIP B:DRAFT.TXT=A:<br>OA>PIP LST:=B:DRAFT.TXT;PASSWORD<br>2B>PIP PRN:=A:DRAFT1.TXT [T81<br>3C>PIP B:ABC.TXT;SECRET[GOI=DEF.TXT;PASS<br>4F>PIP B:ABC.TXT=DRAFT.TXT;PASSWORD [GO]<br>OF>PIP B:=*.*[AVI |
| **PRINTER** n | Displays the printer number for your console. With a number n, it sets the printer number to n for your console.<br>OA>PRINTER<br>3F>PRINTER 2 |

**Table F-1. (continued)**

```
Syntax                Definition and Examples
```

**PRLCOM d:filename.PRL d:filename.Com**

Creates a file of type COM from a file of type PRL. (The file types must be included.)

```
CA>PRLCOM TESTPROC.PRL TESTPROG.COM
```

RDT filespec

Relocatable debugger that debugs assembly language programs of file types COM, HEX or PRL.

```
OA>RDT
3E>RDT A:TESTPROG.PRL
 4F>RDT A:TESTPROG.HEX
1A>RDT TESTPROG.COM
```

REN destination filespec = source filespec

Renames a file without changing its contents. Accepts ambiguous filenames or filetypes if present in both source and destination.

```
VOREN DRAFT.TXT = FIRST.TXT
OA>REN B:DRAFT.TXT = B:FIRST.TXT
6D>REN DRAFT5.TXT = E:DRAFT.TXT;PASSWORD
013>REN *.TEX = *.WRK
```

**RMAC filename** $options

A relocatable assembler, assembling files of type ASM or MAC into REL files which can then be linked to create PRL, RSP, SPR or COM files.

```
OA>RMAC XIOSJMP
0A>RMAC MODULE1 $PP
```

**SCHED** mm/dd/yy hh:mm

When the system date and time equals the time specified the command line is executed.

```
OA>SCHED 02/14/82 12:01 A:CRUN2 MENU
```

**Table F-1. (continued)**

```
Syntax                 Definition and Example

SDIR [global  options] filespec, filespec
                       Displays the disk directory with options.
                            1A>SDIR
                            2A>SDIR B:
                            3B>SDIR [DRIVE=ALL, USER=ALL, SIZE]
                            4C>SDIR [XFCB, DRIVE=(a,b,c))
                            5E>SDIR [USER=2, RW] *.TEX, *.WRK
SET [HELP)
SET d:[NAME=diskname]
SET d:[PASS=password, PROTECT=ON, DEPAULT=password)
SET d:[RO,RW]
SET filespec  [PASS=password,TIME=ON]
SET filespec  [PROTECT=READ,
PROTECT=WRITE,PROTECT=DELETE,PROTECT=NONE]
SET filespec  [RW, RO, DIR, SYS]
                       SET controls password protection, date/time
                       stamps, and file or drive attributes. SET
                       commands affect either an entire drive, a group
                       of files, or a single file. (See Section 7).
                            OA>SET D:[RO]
                            OC>SET *.PRL [RO, SYS], *.COM [RO, SYS]
                            IB>SET *.PRL [SYS,R0,PASS=SECRET,PROT=READ]
                            2C>SET *.COM [RW, PROTECT = NONE, DIR]
                            OA>SET B:[PASSWORD = SECRET]
                            OA>SET [NAME = SYSTMDSK]
                            OA>SET (PASS=<cr>
                            OA>SET *.PRL [PASSWORD = SECRET]
SHOW option,  option, option ...
options = SPACE, USERS, DRIVES, LABEL, HELP
                       Displays amount of free disk space, the drive
                       label status, the active user numbers on a.
                       drive, and the drive characteristics.
                            0A>SHOW
                            OB>SHOW C:
                            lC>SHOW DRIVES
                            2D>SHOW USERS
                            3E>SHOW LABEL
                            3E>SHOW E:,F:,E:USERS,F:USERS
```

**Table F-1. (continued)**

| Syntax | Definition and Example |
|---|---|
| **SPOOL filespec**, filespec | Sends the files specified by filespec to the spool queue, where they await proper handling by the list device.<br>0A>SPOOL **DOCUMENT.LAW, B:DOCUMENT.TXT** |
| **STAT d:=RO**<br>**STAT d:DSK:**<br>**STAT d:USR:**<br>**STAT VAL:**<br>**STAT** filespec [attribute]<br>attributes = RO, RW, SYS, DIR or SIZE | Provides information about a file or a group of files on a disk. Also assigns attributes to a file, disk, or drive. The SIZE attribute displays the computed file size of the file or files.<br>0A>STAT<br>0A>STAT B:<br>1F>STAT D:*.PRL |
| **STOPSPLR STOPSPLR n** | Stops the, spooling operation currently in progress, initiated from console number n.<br>0A>STOPSPLR |
| **SUBMIT filename** [actual parameters] | SUBMIT submits a batch process consisting of a file of MP/M II commands (one command per line in file). The. filename must be a file of type SUB. Parameters following the filename are substituted for their corresponding parameters in the file. (See Section 10.6)<br>OA>SUBMIT START.SUB<br>OA>SUBMIT B:START.SUB<br>OA>SUBMIT START.SUB B LETTER<br>1F>SUBMIT B:START.SUB |

**Table F-1. (continued)**

| Syntax | Definition and Examples |
|---|---|
| **TOD** PERPETUAL<br>**TOD mm/dd/yy hh:mm:ss** | TOD displays the system date and time. with a date and time, TOD sets the system time to the date and time specified. The PERPETUAL option displays date and time continuously. (it can be abbreviated P).<br>OA>TOD<br>1B>TOD P<br>1C>TOD 02/14/82 12:01:00 |
| **TYPE filespec** (PAGE] | Displays contents of a file containing ASCII coded information. Does not accept ambiguous filenames. The [PAGE] option pauses after each screen (24 lines) is displayed until you strike a key. [Pn] specifies the number of lines per screen.<br>OA>TYPE DRAFT.TXT<br>OA>TYPE B:DRAFT.TXT [PAGE]<br>1F>TYPE E:DRAFT.TXT;PASSWORD[P15] |
| **USER** n | Changes the current user number. Sets the user number to n, where n is an integer from 0-15.<br>OA>USER 8 |
| **XREF filename** $P | An assembly language cross-reference utility that can be applied to the PRN and SYM files produced by RMAC in order to provide a summary of variable usage throughout the program. The filename must be the name of both the PRN and SYM files. A file of type XRF is output. The $P option directs output to the list device (printer).<br>OA>XREF MAINPROG<br>OA>XREF MAINPROG $P |

# Appendix G
# CP/M and MP/M Comparative Command Summary

In the following table, X means the utility is supplied with the system. * means the utility can be purchased separately.

Table G-1. Comparative Command Summary

| Command | CP/M 1.4 | CP/M 2 | MP/M 1 | MP/M II |
|---|---|---|---|---|
| ABORT | | | X | X |
| ASM | X | X | X | X |
| ATTACH | | | X | X |
| CONSOLE | | | X | X |
| DDT | X | X | X | X |
| DIR | X | X | X | X |
| DSKRESET | | | X | X |
| DUMP | X | X | X | X |
| ED | X | X | X | X |
| ERA | X | X | X | X |
| ERAQ | | | X | X |
| GENHEX | X | X | X | X |
| GENMOD | | | X | X |
| GENSYS | | | X | X |
| LIB | | * | * | X |
| LINK | | * | * | X |
| LOAD | X | X | X | X |
| MOVCPM | X | X | | |
| MPMLDR | | | X | X |
| MPMSTAT | | | X | X |
| PIP | X | X | X | X |

**Table G-1. (continued)**

| Command | CP/M 1.4 | CP/M 2 | MP/M 1 | MP/M II |
| --- | --- | --- | --- | --- |
| PRINTER | | | x | x |
| PRLCOM | | | x | x |
| RDT | | | x | x |
| REN | x | x | x | x |
| RMAC | | | | x |
| SAVE | x | x | | |
| SCHED | | | x | x |
| SDIR | | | | x |
| SET | | | | x |
| SHOW | | | | x |
| SPOOL | | | x | x |
| STAT | x | x | x | x |
| STOPSPLR | | | x | x |
| SUBMIT | x | x | x | x |
| SYSGEN | x | x | | |
| TOD | | | x | x |
| TYPE | x | x | x | x |
| USER | | x | x | x |
| XREF | | | | x |
| XSUB | | x | | |

# Appendix H
# Drive and File Status Summary

**Table H-1. Display File Size**

| Command | # of Bytes In File | # of Recs in File | File Size | Free Disk Space | Totals |
|---|---|---|---|---|---|
| SDIR | X | X | | X | X |
| SDIR [SIZE] | X | | | | |
| STAT filespec | X | X | | X | X |
| STAT filespec [SIZE] | X | X | X | X | X |

**Table H-2. Display File Attributes**

| Command | SYS DIR | RW RO | SYS Files displayed | User # | Archive |
|---|---|---|---|---|---|
| DIR [SYS] | | | X | | |
| DIR filespec [SYS] | | | X | | |
| SDIR | X | X | X | X | X |
| STAT filespec | | X | X | | X |
| STAT filespec[SIZE] | X | X | X | X | X |

**Table H-3. Display Time Stamping and Protection Modes**

| Command | Creation or Last Access | Last Update | Password Mode | XFCB |
|---|---|---|---|---|
| SDIR | X | X | X | X |

**Table H-4. Ways to Display Disk, Label, and System Status**

| Command | RO RW | Free Disk Space | STAT Display | Active Users | Active Files | Label Display |
|---|---|---|---|---|---|---|
| SHOW | X | X | | | | |
| SHOW d: | X | X | | | | |
| SHOW Space | X | X | | | | |
| SHOW DRIVE | | | X | | | |
| SHOW USERS | | | | X | X | |
| SHOW LABEL | | | | | | X |
| STAT | X | X | | | | |
| STAT d: | X | X | | | | |
| STAT [SIZE] | X | X | | | | |
| STAT USER | | | | X | X | |
| STAT DSK: | | | X | | | |

**Sample SHOW DRIVE Display:**

**OA>SHOW DRIVE**

| | |
|---|---|
| B: | Drive Characteristics |
| 65,536: | 128 Byte Record Capacity |
| 8,192: | Kilobyte Drive Capacity |
| 512: | 32 Byte Directory Entries |
| 0: | CheckedDirectory Entries |
| 1,024: | Records Directory Entry |
| 128: | Records Block |
| 68: | Sectors Track |
| 0: | Reserved Tracks |

**Sample SHOW LABEL Display:**

| Directory Label | Passwds Req'd | Make XFCBs | Stamp | Stamp Create | Label Update | Created Label Updated |
|---|---|---|---|---|---|---|
| TOMSDISK.DAT | on | on | on | on | 07/04/81 10:30 07/08/81 09:30 | |

# Appendix I
# User's Glossary

**ambiguous filename**: Filename that contains either of the MP/M II wildcard characters, ? or *, in the primary filename or the filetype or both. When you replace characters in a filename with these wildcard characters, you create an ambiguous filename and can easily reference more than one MP/M II file in a single command line. See Section 3 of this manual.

**applications program**: Program that needs an operating system to provide an environment in which to execute. Typical applications programs are business accounting packages, word processing (editing) programs, mailing list programs, etc.

**archive attribute**: File attribute that indicates whether or not the file has been backed up. When you use PIP with the Archive option, it turns the archive attribute on. When a program changes a file, MP/M II turns the archive attribute off, indicating that the file is new, and not backed up.

**argument**: Symbol, usually a letter, indicating a place into which you can substitute a number, letter or name to give an appropriate meaning to the formula in question.

**ASCII**: The American Standard Code for Information Interchange is a standard code for representation of numbers, letters, and symbols. An ASCII text file is a file that can be intelligibly displayed on the video screen or printed on paper. See Appendix A. **attribute**: File characteristic that can be set to on or off.

**back-up**: Copy of a disk or file made for safe keeping, or the creation of this disk or file.

bit: "Switch" in memory that can be set to on (1) or off (0) Bits are grouped into bytes.
**block**: Area of memory or disk reserved for a specific use.

**bootstrap**: Process of loading+ an operating system into memory. Bootstrap procedures vary from system to system. The boot for an operating system must be customized for the memory size and hardware environment that the operating system will manage. Typically, the boot is loaded automatically and executed at power up or when the computer is reset. Sometimes called a "cold start."

buffer: Area of memory that temporarily stores data during the transfer of information.

**built-in commands**: Commands that permanently reside in memory. They respond quickly because they are not accessed from a disk.

**Byte:** unit of memory or disk storage containing eight bits.

**command:** Elements of an MP/M II command line. In general, an MP/M II command has three parts: the command keyword, the command tail, and a carriage return.

**command file:** Series of coded machine executable instructions stored on disk as a program file, invoked in MP/M II by typing the command keyword next to the system prompt on the console. The MP/M II command files generally have a filetype of PRL or COM. Files are either command files or data files. Same as a command program.

**command keyword:** Name that identifies an MP/M II command, usually the primary filename of a file of type COM or PRL, or the name of a queue associated with a Resident System Process. The command keyword precedes the command tail and the carriage return in the command line.

**command syntax:** Statement that defines the correct way to enter a command. The correct structure generally includes the command keyword, the command tail, and a carriage return. A syntax line usually contains symbols that you should replace with actual values when you enter the command.

**command tail:** Part of a command that follows the command keyword in the command line. The command tail can include a drive specification, a filename and/or filetype, a password, and options or parameters. Some commands do not require a command tail.

**oncatenate:** Term that describes one of PIP's operations that copies two or more separate files into one new file in the specified sequence.

**console:** Primary input/output device. The console consists of a listing device such as a screen and a keyboard through which the user communicates with the operating system or applications program. A console is a terminal that is capable of initiating programs. MP/M II refers to these as system consoles.

**control character:** Non-printing character combination that sends a simple command to the currently executing process. To enter a control character, hold down the CONTROL key on your terminal and strike the character key specified. In this document, the CONTROL key is represented by an up-arrow: T. A TX, for example, erases the command line. See Appendix C.

**cursor:** One-character symbol that can appear anywhere on the console screen. The cursor indicates the position where the next keystroke at the console will have an effect.

**data file:** Non-executable collection of similar information that generally requires a command file to manipulate it.

**default:** Currently selected disk drive, user number or password. Any command that does not specify a disk drive or a user number references the default disk drive and user number. When MP/M II is first invoked, the default disk drive is drive A, and the default user number varies from console to console, until changed with the USER command. A default display is a display generated by a command keyword without any options.

**delimiter:** Special characters that separate different items in a command line. For example, in MP/M II, a colon separates the drive specification from the filename. A period separates the filename from the filetype. A semicolon separates the filename and type from the password, and square brackets separate any options from their command or file specification. Commas separate one item in an option list from another. All of the above special characters are delimiters.

**directory:** Portion of a disk that contains entries for each file on the disk. In response to the DIR command, MP/M II displays the filenames stored in the directory.

**DIR attribute:** File attribute that causes a file to be accessible from the default user number and drive only. **directory label:** Same as label.

disk, diskette: Magnetic media used to store information. Programs and data are recorded on the disk in the same way that music is recorded on a cassette tape. The term "diskette" refers to smaller capacity removable floppy diskettes. "Disk" can refer to a diskette, a removable cartridge disk or a fixed hard disk.

**disk drive:** Peripheral device that reads and writes on hard or floppy disks. MP/M II assigns a letter to each drive under its control. For example, MP/M II may refer to the drives in a four drive system as A, B, C, and D. **drive label:** Same as label.

**editor:** Utility program that creates and modifies text files. An editor can be used for creation of documents or creation of code for computer programs. The MP/M II editor is invoked by typing the command ED next to the system prompt on the console. (See ED in Section 12 of this manual).

**executable:** Ready to be run by the computer. Executable code is a series of instructions that can be carried out by the computer. For example, the computer cannot "execute" names and addresses, but it can execute a program that prints all those names and addresses on mailing labels. FCB: File Control Block.

**field:** Portion of a record containing one data item such as a person's name, an address, or a phone number.

**file:** Collection of characters, instructions or data stored on a disk. The user can create files on a disk.

**File Control Block:** Structure used for accessing files on disk. Contains the drive, filename and filetype of a file to be accessed or created on the disk.

**filename:** Name assigned to a file. A filename can include a primary filename of 1-8 characters and a filetype of 0-3 characters. A period separates the primary filename from the filetype.

**file specification:** Unique file identifier. A complete MP/M II file specification includes a disk drive specification followed by a colon (d:) , a primary filename of 1 to 8 characters, a period and a filetype of 0 to 3 characters, a semicolon and a password. For example, b: example.tex;password is a complete MP/M II file specification.

**filetype:** Extension to a filename. A filetype can be from 0 to 3 characters and must be separated from the primary filename by a period. A filetype can tell something about the file. Certain programs require that files to be processed have certain filetypes (see Appendix B).

**floppy disk:** Flexible magnetic disk used to store information. Floppy disks come in 5 1/4 and 8 inch diameters.

**hard disk**: Rigid, platter-like, magnetic disk sealed in a container. A hard disk stores more information than a floppy disk.

**hardware:** Physical components of a computer.

**hex file**: ASCII-printable representation of a command (machine language) file.

**hexadecimal notation:** Notation for the base 16 number system using the symbols 0,1,2,3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F to represent the sixteen digits. Machine code is often converted to hexadecimal notation because it can be easily represented by ASCII characters and therefore printed on the console screen or on paper (see Appendix A).

**input:** Data going into the computer, usually from an operator typing at the terminal or by a program reading from the disk.

**Interface:** Object that allows two independent systems to communicate with each other, as an interface between hardware and software in a microcomputer. I/0: Abbreviation for input/output.

**keyword:** See **command keyword.**

**kilobyte**: 1024 bytes denoted as 1K. 32 kilobytes equal 32K. 1024 kilobytes equal one megabyte, or over one million bytes.

**label**: Entry within the directory. The optional label contains information that describes special attributes of the disk to the operating system. For example, the label tells MP/M II whether or not time stamping and password protection are turned on for that disk. You can give a label a name to help identify the data that is stored on a given disk.

**list device**: Device such as a printer onto which data can be listed or printed.

**logged in**: Made known to the operating system, in reference to drives. A drive is logged in when it is selected by the user or an executing process, and remains selected or logged in until a DSKRESET is performed or the whole operating system is reset.

**logical**: Representation of something that may or may not be the same in its actual physical form. For example, a hard disk can occupy one physical drive, and yet you can divide the available storage on it to appear to the user as if it were in several different drives. These apparent drives are the logical drives. megabyte: Over one million bytes; 1024 kilobytes (see byte).

**microprocessor**: Silicon chip that is the Central Processing Unit (CPU) of the microcomputer.

**MP/M II**: Multi-Programming Monitor Control Program for microprocessors.

**multi-prog ramming**: Capability of the operating system to coordinate the execution of more than one program at a time.

**multi-user**: Ability of an operating system to support more than one independent user initiating different programs at the same time.

**operating system**: Collection of programs that supervises the running of other programs and the management of computer resources. An operating system provides an orderly input/output environment between the computer and its peripheral devices. It enables user written programs to execute safely.

**option**: One of many variables that can be appended to a command.

**output**: Data that the processor sends to the console, disk or printer.

**page relocatable**: Program file that can be loaded into any available memory segment.

**parameter:** Value in the command tail that provides additional information for the command. Technically, a parameter is a required element of a command.

**password:** User-created extension to a filename that enables the user to add extra protection to his files. The password may then be required to access that file. A password can be up to eight characters long and include any numeric or upper- or lower-case characters and some special characters.

**peripheral devices:** Devices external to the CPU. For example, terminals, printers and disk drives are common peripheral devices that are not part of the processor, but are used in conjunction with it.

**physical:** Actual hardware of a computer. The physical environment varies from computer to computer.

**primary filename:** First 8 characters of a filename. The primary filename is a unique name that helps the user identify the file contents. A primary filename contains 1 to 8 characters and can include any letter or number and some special characters. The primary filename follows the optional drive specification and precedes the optional filetype.

**process:** When a program is actually executing, as opposed to being in a static state of storage on disk, it is called a process.

**program:** Series of specially coded instructions that performs specific tasks when executed by a computer.

**prompt:** Any characters displayed on the video screen to help the user decide what the next appropriate action is. A system prompt is a special prompt displayed by the operating system. The System prompt indicates to the user that the operating system is ready to accept input. The MP/M II system prompt is two characters followed by an angle bracket. The first character is numeric and indicates the default user number. The, second character is alpha and indicates the default drive. Some applications programs have their own special "system" prompts.

**queue:** First in, first out list. Under MP/M II, a queue is treated as a file in memory. For example, a queue has a Queue Control Block instead of a File Control Block. In general, processes use queues to pass information to other processes. one of the functions of a queue is to "wake up" a Resident System Process and pass your command to it. Refer to the MP/M II Programmer's Guide for further discussion.

**Read Only:** Attribute that can be assigned to a disk file or a disk drive. When assigned to a file, the Read only attribute allows you to read from that file but not write any changes to it. When assigned to a drive, the Read Only attribute allows you to read any file on the disk, but prevents you from adding a new file, erasing or changing a file, renaming a file, or writing on the disk. The

SET and STAT commands can set a file or a drive to Read Only. Every file and drive is either Read Only or Read Write. The default setting for drives and files is Read Write, but an error in setting disk density or resetting the disk or other error found by MP/M II automatically sets the drive to Read Only until the error is corrected. Files and disk drives may be set to either Read Only or Read Write.

**Read Write:** Attribute that can be assigned to a disk file or a disk drive. The Read Write attribute allows you to read from and write to a specific Read Write file or to a any file on a disk that is in a drive set to Read Write. A file or drive can be set to either Read Only or Read Write.

**record:** Collection of data. A file consists of one or more records stored on disk. An MP/M II record is 128 bytes long. **relocatable program:** Same as page relocatable program.

**Resident System Process:** A process that is made part of MP/M II during system generation. An RSP can be associated with a queue that can be accessed by a command keyword, but does not exist on disk as a command file. RO: Abbreviation for Read Only. RSP: Resident System Process.

**run a program:** Start a program executing. When a program is running, the computer is executing a sequence of instructions. RW: Abbreviation for Read Write.

**sector:** Portion of a disk track. There are a specified number of sectors on each track.

**software:** Specially coded programs transmit machine readable instructions to the computer, as opposed to hardware, which is the actual physical components of a computer.

**source file:** ASCII text file that is an input file for a processing program, such as an editor, text formatter, or assembler.

**spooling:** Printing a file from disk. The SPOOL program, which is detached from a console, can print a file from a disk. This leaves your console free for other tasks while your file is being printed.

**syntax:** Format for entering a given command.

**system attribute:** Attribute assigned to a file enabling that file to be accessed from other than the default drive and user number. System files are generally placed on the system drive in user 0 because they are most easily and efficiently accessed from that location. A file can have either the SYS attribute or the DIR attribute.

**system console:** Same as console.

**system drive:** Drive on which MP/M II looks for files with the SYS attribute after checking the default or specified drive. The system drive is specified at system generation.

**systems program:** Program that contributes to the operating system package.

**system prompt:** Symbol displayed by the operating system indicating that the system is ready tc receive input. See prompt.

**terminal:** Generally, the same as a console. However, some terminals are not capable of initiating programs, and these terminals cannot be considered consoles. MP/M II supports up to eight system consoles and up to eight character input/output terminals that are not consoles.

**time-stamp:** Record of when a file was created, accessed or updated. In MP/M II, the SET command turns time-stamp on (see Section 7). The time and date information is appended to a file in the XFCB. The time stamps are displayed by the SDIR command described in Section 6 of this manual.

**track:** Concentric rings dividing a disk. There are 77 tracks on a typical eight inch floppy disk.

**turn-key application:** Application designed for the non computer oriented user. For example, a typical turn-key application is designed so that the operator needs only to turn on the computer, insert the proper program disk and select the desired procedure from a selection of functions (menu) displayed on the screen.

**upward-compatible:** Term meaning that a program created for the previously released operating system (or compiler, etc.) runs under the newly released version. of the same operating system. For example, MP/M II is an upward-compatible version of MP/M I because programs that ran under MP/M 1 can run properly under MP/M I!.

**user number:** Number assigned to a region of the disk directory so that different users need only deal with their own files and have their "own" directories even though they are all working from the same disk. In MP/M II, there can be up to sixteen users on a single disk.

**utility:** "Tool." Program that enables the user to perform certain operations, such as copying files, erasing files, and editing files. Utilities are created for the convenience of programmers and users. MP/M II is distributed with over thirty utilities.

**wildcard characters:** Special characters that match certain specified items. In MP/M II there are two wildcard characters, ? and *. The ? can be substituted for any single character in a filename, and the * can be substituted for the primary filename or the filetype or both. By placing wildcard characters in filenames, the user creates an ambiguous filename and can quickly reference one or more files.

**XFCB:** Extended File Control Block. XFCBs store passwords, and time and date stamping of files. See FCB and Block.

# Index

U



W



x