CPMPUG6.WS4 (= "CP/M-86 Plus User's Guide", section 6)

(Edited by Emmanuel ROCHE.)

Section 6: Command Reference Section

This section describes the commands supplied with CP/M-86 Plus. Command descriptions are in alphabetical order. Each description includes the options that you can use with the command, a short explanation of the use for the command, and practical examples of how the command is used. The commands are preceded by a description of the command format to use when you invoke these commands, and the notation used in the command descriptions.

6.1 USING CP/M-86 PLUS COMMANDS

As described in Section 4, the CP/M-86 Plus command consists of a command keyword and an optional command tail. The command tail can include parameters and other options. A file specification can be included as one of the parameters. A file specification names a particular file or group of files in the directory of an on-line disk. For example,

    B:MYFILE.A86

is a file specification that indicates drive B, filename MYFILE, and filetype A86. File specification is abbreviated as "filespec" in the command format statements. It includes four parts:

1) drive specifier
   Optional disk drive, A, B, ... P that contains the file or group of files to which you are referring. If a drive specifier is included in your filespec, it must be followed by a colon (":").

2) filename
   One- to eight-character first name of a file or group of files.

3) filetype
   Optional one- to three-character family name of a file or group of files. If the filetype is present, it must be separated from the filename by a period (".").

4) password
   Optional one- to eight-character password that allows you to protect your files. It follows the filetype (or the filename if no filetype is assigned) and is preceded by a semicolon (";").

A file specification has the following general form:

    d:filename.typ;password

In this form, D: represents the optional drive specification, FILENAME represents the one- to eight-character filename, .TYP represents the optional one- to three-character filetype, and ;PASSWORD represents the optional one-to eight-character password. The following are valid combinations of the elements of a CP/M-86 Plus file specification.

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

If you do not include a drive specifier, CP/M-86 Plus commands automatically use the default drive. If you omit the period (".") and the filetype, CP/M-86 Plus commands automatically use a filetype of three blanks.

Some CP/M-86 Plus commands accept wildcards in the filename and filetype parts of the command tail. For example,

B:MY*.A??

is a file specification that indicates drive B, filename MY*, and filetype A??. This file specification might match several files in the directory.

6.2 HOW CP/M-86 PLUS COMMANDS ARE DESCRIBED

CP/M-86 Plus commands appear in alphabetical order. Each command description is given in a specific form. This section also describes the notation that indicates the optional parts of a command tail and other syntax notation.

- The description begins with the command keyword in uppercase, followed by the command parameters and options.

- The syntax section uses the notation described below, and gives one or more general forms to follow when you compose the command line.

- The explanation section defines the general use of the command keyword, and points out exceptions and special cases. The explanation sometimes includes tables or lists of options that you can use in the command line.

- The examples section lists a number of valid command lines that use the command keyword.

The notation in the syntax lines describes the general command form, using these rules:

- Words in capital letters must be spelled as shown, but you can use any combination of upper- or lowercase letters.

- The symbolic notation d:, filename, .typ, ;password, and filespec have the general meanings previously described.

- In some commands, descriptive qualifiers are used with filespecs to further qualify the type of filespec accepted by the commands. For example, wildcard-filespec denotes wildcard specifications, dest-filespec denotes a destination filespec, and src-filespec denotes a source filespec.

- You must include one or more space characters where a space is shown, unless otherwise specified.

The following table defines the special symbols and abbreviations used in syntax lines.

Table 6-1. Syntax Notation

Symbol	Meaning
DIR	Directory attribute.
n	You can substitute a number for n.
o	Indicates an option or an option list.
RO	Read-Only.
RW	Read/Write.
s	You can substitute a string, which consists of a group of characters, for s.
SYS	System attribute.
{}	Items within braces are optional. You can enter a command without the optional items. The optional items add features to your command line.
[]	Items in square brackets are options or an option list. If you use an option specified within the square brackets, you must type the left and right square brackets ("[" and "]") to enclose the option. If the right square bracket ("]") is the last character on the command line, it can be omitted.
()	Items in parentheses indicate a range of options. If you use a range from an option list, you must enclose the range in parentheses.
...	Ellipses tell you that the previous item can be repeated any number of times.
|	OR bar separates alternative items in a command line. You can select any or all of the alternatives specified. Mutually exclusive options are indicated in additional syntax lines, or are specifically noted in the text.
^ or Ctrl	Represent the Ctrl key on your keyboard. (Control characters appear as ^ on your screen.)
ENTER	Indicates a carriage return keystroke.
*	Wildcard character -- replaces all or part of a filename and/or filetype.
?	Wildcard character -- replaces any single character in the same position of a filename or filetype.

Look at an example of syntax notation. The CP/M-86 Plus command PIP

(Peripheral Interchange Program) is the file copy program. PIP can copy information from your screen to the disk or printer. PIP can combine two or more files into one longer file. PIP can also rename files after copying them. The following is an example of one of the formats of the PIP command line that shows how to use command line notation.

Syntax: PIP dest-filespec{[Gn]}=src-filespec{[Gn]}

For this example, dest-filespec is further defined as a destination file specification, or peripheral device (printer, for example) that receives data. Similarly, src-filespec is the source file specification, or a peripheral device (keyboard, for example) that transmits data. The Gn option indicates in which user number the source filespec can be found, and to which user number it is sent. PIP accepts wildcards in the filename and filetype. (See the PIP command summary for details regarding other capabilities of PIP.) There are many valid command lines that come from this format. For example,

```
PIP NEWFILE.DAT=OLDFILE.DAT
PIP B:=A:THISFILE.DAT
PIP B:X.TXT=Y.TXT,Z.TXT
PIP X.TXT=A.TXT,B.TXT,C.TXT
PIP B:=A:*.BAK
PIP B:=A:*.*
```

The BACK Command

Syntax: BACK filespec [command tail] {[INFILE=filespec,OFILE=filespec]}
{[NOBAC,RSX=(filespec{,filespec,...})]}

Explanation: The BACK command places a program in the background, allowing you to use the terminal for other work. The background program must be a command (CMD) file. If the command file requires a command tail, then it immediately follows the command file specification.

A program that executes in the background does not have access to the console. Console input must be supplied using the INFILE option, which specifies the input redirection file specification. Normally, a background task is assumed to have no console input. If the program attempts to read input from the console, and an INFILE is not specified or does not contain all of the required input, it is terminated. BACK automatically redirects console output to a file called XXX.BAC, where XXX is the program ID number associated with the background program. The BAC file contains all the output that normally is seen on the console. Any messages or error conditions are recorded in the BAC file. The output file can be named something other than XXX.BAC by using the OFILE option.

The NOBAC option tells BACK not to create a BAC file, and not to save any of the console output. If this option is used, you cannot determine what happened if a background job is terminated abnormally.

The RSX option allows you to load an RSX that is used by the program. Each filespec in the option has a filetype of RSX, and is not attached to the

command file. These RSXs are loaded in the RSX chain following any console redirection RSXs. See Section 5 for a discussion of background tasking.

Example: A>BACK PIP LST:=b:memos.old

Puts PIP.CMD in the background. The command tail, LST:=b:memos.old, tells PIP to copy the file B:MEMOS.OLD to the printer, LST:. A console output file is created that contains any console output from PIP during this run.

A>BACK SUBMIT WORK [INFILE=address.lis,OFILE=mail.bac]

Runs SUBMIT.CMD in the background with WORK.SUB as input to the SUBMIT command. Console input to WORK.SUB is provided by the file ADDRESS.LIS. The console output file is named MAIL.BAC.

A>BACK LINK [LO] MYPROG [NOBAC]

LINK.CMD is executed in the background with options L and O. MYPROG.OBJ is input to LINK. No console output file is created.

```
C>submit demoback

C>back pip lst:=temp.txt [ofile=temp1]

Background output in file: TEMP1
C>back pip lst:=temp.txt [ofile=temp2]

Background output in file: TEMP2
C>back pip lst:=temp.txt [ofile=temp3]

Background output in file: TEMP3
C>

C>task
0:  PIP
1:  PIP
2:  PIP
3:  TASK
Stop which program? ENTER     (=Press the ENTER key on your keyboard.)

C>show [program]
0:  PIP
1:  PIP
2:  PIP
3:  SHOW
```

The DATE Command

Syntax: DATE {CONTINUOUS}
DATE {time-specification}
DATE SET

Explanation: The DATE command allows you to display and set the date and time of day. When you start CP/M-86 Plus, the date and time are set to the creation date of your CP/M-86 Plus system. Use DATE SET to change this initial value to the current date and time.

Set the Date and Time

Syntax: DATE {time-specification}
DATE SET

Explanation: The first form allows the user to enter both date and time in the command line. The time-specification has the format:

MM/DD/YY HH:MM:SS

where:

MM is the month value in the range 1 to 12.
DD is the day value in the range 1 to 31.
YY is the two-digit year value relative to 2000.
HH is the hour value in the range of 0 to 23.
MM is the minute value in the range of 0 to 59.
SS is the second value in the range of 0 to 59.

The system checks the validity of the date and time entry, and determines the day for the date entered. Note that the system uses a 24-hour clock.

The second form prompts you to enter the date and the time. To keep the current system date or time, press ENTER.

Examples: A>DATE 08/14/04 10:30:0

The system responds with the following:

Press any key to set time

When the time is displayed, press any key. DATE initializes the time at that instant, and displays the date and time:

Sat 08/14/04 10:30:00

A>DATE SET

The system prompts with the following:

Enter today's date (MM/DD/YY):

Enter the date, then press ENTER. Then the system prompts with the following:

Enter the time (HH:MM:SS):

Enter the time and press ENTER, and the system prompts with the following:

Press any key to set time

You can set the time exactly. Pressing only ENTER after each prompt allows you to skip the date and time set process.

Display Current Date and Time

Syntax: DATE CONTINUOUS

Explanation: The preceding form of the DATE command displays the current date and time. The CONTINUOUS option allows continuous display of the date and time. The CONTINUOUS option can be abbreviated to C. You can stop the continuous display by pressing any key.

Examples: A>DATE
A>DATE C

The first example displays the current date and time. The following is a sample display:

Fri 08/13/02 09:15:37

The second example displays the date and time continuously, until you press any key to stop the display.

The DEVICE Command

Syntax: DEVICE {NAMES | VALUES | physical-dev | logical-dev}
DEVICE log-dev=phys-dev {option} {phys-dev {option}...,}
DEVICE log-dev=NULL
DEVICE phys-dev {option}
DEVICE CONSOLE [PAGE | COLUMNS = # | LINES = #]

Explanation: The DEVICE command displays current assignments of CP/M-86 Plus logical devices, and the names of physical devices. DEVICE allows you to assign logical CP/M-86 Plus devices to peripheral devices attached to the computer. The DEVICE command also sets the communication protocol, data bits, stop bits, parity, polarity, and speed. The DEVICE command also displays the current device characteristics, and sets the current console screen size.

CP/M-86 Plus supports the following five logical devices:

CONIN:
CONOUT:
AUXIN:
AUXOUT:
LST:

These logical devices are also known by the following names:

CON: (for CONIN: and CONOUT:)
CONSOLE: (for CONIN: and CONOUT:)
AUX: (for AUXIN: and AUXOUT:)
AUXILIARY: (for AUXIN: and AUXOUT:)

The physical device names on a computer vary from system to system. You can use the DEVICE command to display the names and attributes of the physical devices that your system accepts.

Display Device Characteristics and Assignments

Syntax: DEVICE {NAMES | VALUES | physical-dev | logical-dev}

Explanation: The preceding form of the DEVICE command displays the names and attributes of the physical devices, and the current assignments of the logical devices in the system.

Examples:

```
A>DEVICE
Physical devices
----------------
Type         :(I=Input, O=Output, S=Serial)
Pr = Protocol :(X=Xon/Xoff, E=ETX/ACK, R=RTS, D=DTR)
Po = Polarity :(H=High, L=Low)
Pa = Parity   :(O=Odd, E=Even, M=Mark, S=Space)
D            : Databits
S            : Stopbits

Name   BAUD(tx,rx) Type  PrPoPaD S     Name   BAUD(tx,rx) Type  PrPoPaD S
------ ----------- ----- - - - - ---   ------ ----------- ----- - - - - ---
CON0   NONE ,NONE  I O P       8 1     AUX0   9600 ,9600  I O S       8 1
AUX1   9600 ,9600  I O S       8 1     PRN0   NONE ,NONE    O P       8 1

Current Assignments:
CONIN:  = CON0
CONOUT: = CON0
AUXIN:  = AUX0
AUXOUT: = AUX0
LST:    = PRN0

Enter new assignment or hit ENTER
```

The preceding command displays the physical devices and current assignments of the logical devices in the system. The system prompts for a new device assignment. You can enter any valid device assignment (as described in the next section). If you do not want to change any device assignments, press ENTER.

A>DEVICE NAMES

Physical devices

Type :(I=Input, O=Output, S=Serial)
Pr = Protocol :(X=Xon/Xoff, E=ETX/ACK, R=RTS, D=DTR)
Po = Polarity :(H=High, L=Low)
Pa = Parity :(O=Odd, E=Even, M=Mark, S=Space)
D : Databits
S : Stopbits

Name	BAUD(tx,rx)	Type	PrPoPaD	S	Name	BAUD(tx,rx)	Type	PrPoPaD	S
CON0	NONE ,NONE	I O	P	8	1	AUX0	9600 ,9600	I O	S	8	1
AUX1	9600 ,9600	I O	S	8	1	PRN0	NONE ,NONE	O	P	8	1

The preceding command lists the physical devices with a summary of the current device characteristics.

A>DEVICE VALUES

Current Assignments:
CONIN: = CON0
CONOUT: = CON0
AUXIN: = AUX0
AUXOUT: = AUX0
LST: = PRN0

The preceding command displays the current logical to physical device assignments.

A>DEVICE CON0

Physical Device: CON0
Baud Rate: NONE ,NONE
Characteristics: INPUT
OUTPUT
PARALLEL
its <--- Yes, it is a bug!

The preceding command displays the assignment of the logical device CON0.

Assign a Logical Device

Syntax: DEVICE logical-dev=physical-dev{option} {physical-dev{option},...}
DEVICE logical-dev=NULL

Explanation: The first form assigns a logical device to one or more physical devices. The second form disconnects the logical device from any physical device.

Examples: A>DEVICE CONOUT:=LPT,CRT

The previous command assigns the system console output, CONOUT:, to the printer, LPT, and the screen, CRT.

```
A>DEVICE AUXIN:=CRT2 [PROTOCOL=XON, SPEED=(9600,9600)]
```

The previous command assigns the auxiliary logical input device, AUXIN:, to the physical device CRT2, using XON/XOFF protocol, and transmission and receiving rates at 9600 baud.

```
A>DEVICE LST:=NULL
```

The previous example disconnects the list output device, LST:.

Set Attributes of a Physical Device

Syntax: DEVICE physical-dev {option}

Explanation: The preceding form of the DEVICE command sets the attributes of the physical device specified in the command.

Example: A>DEVICE LPT [PROTOCOL=XON, SPEED=(9600,9600)]

The preceding command sets the XON/XOFF protocol for the physical device, LPT, and sets the transmission and receiving rates to 9600 baud.

Display or Set the Current Console Screen Size

Syntax: DEVICE [PAGE | COLUMNS=# | LINES=#]

Explanation: The preceding form of the DEVICE command displays or sets the current console size.

Examples:

```
A>DEVICE CONSOLE [PAGE]

Console width set to 80 columns
Console page set to 24 lines

A>DEVICE CONSOLE [COLUMNS=64, LINES=16]

Console width set to 16 columns <--- Yes, this is a bug!
Console page set to 24 lines    <--- And another bug...
```

The first example displays the current console page width in columns and length in lines. The second example sets the screen size to 64 columns and 16 lines.

Table 6-2. DEVICE Options

Format: Option
Meaning

DAT{ABITS}=5 | 6 | 7 | 8
datum consists of 5, 6, 7, or 8 data bits.

PAR{ITY}=NONE | ODD | EVEN | MARK | SPACE
specifies what type of parity this device uses.

POL{ARITY}=HIGH | LOW
specifies what type of polarity for this device.

PRO{TOCOL}=NONE | XON | ETX | RTS | DTR
specifies what protocol this device is using for communication.

SP{EED}=(tx{,rx})
specifies the data transmission rate for the device. If the transmission rate only, tx, is specified then the receiving rate, rx, is assumed to be the same. tx and/or rx must be one of the following:

NONE 50 62 75
110 134 150 200
300 600 1200 1800
2000 2400 3600 4800
7200 9600 192 384
560 768 OEM1 OEM2
OEM3

OEM number is determined by the system implementor for your hardware.

ST{OPBITS}=1 | 15 | 2
specifies how many stop bits are sent with each character.

The DIR Command

Syntax: DIR {d:}
DIR {filespec}

DIRSYS {d:}
DIRSYS {filespec}

DIR {d:} [options]
DIR {filespec} {filespec}...[options]
DIR [options] {filespec,filespec...}

Explanation: The DIR (Directory) command displays the names of files, and the attributes associated with the files. DIR and DIRSYS are built-in utilities; DIR with options is a transient utility.

Display Directory

Syntax: DIR {d:}
DIR {filespec}

DIRSYS {d:}
DIRSYS {filespec}

Explanation: The DIR and DIRSYS commands display the names of files catalogued in the directory of an on-line disk. DIR and DIRSYS both accept wildcards in the file specification.

DIRSYS can be abbreviated to DIRS.

The DIR command lists the names of files in the current user number that have the DIR attribute. The DIRSYS command displays the names of files in the current user number that have the System (SYS) attribute. Although you can read System files that are stored in user 0 from any other user number on the same drive, DIRSYS only displays user 0 files if the current user number is 0.

If you omit the drive and file specifications, the DIR command displays the names of all files with the DIR attribute on the default drive for the current user number. Similarly, DIRSYS displays all the SYS files.

If the drive specifier is included, but the filename and filetype are omitted, the DIR command displays the names of all DIR files in the current user on the disk in the specified drive. DIRSYS displays the SYS files.

If the file specification contains wildcard characters, all filenames that satisfy the match are displayed on the screen.

If SYS files match the file specification, DIR displays the message:

SYSTEM FILE(S) EXIST

If non-system (DIR) files match the file specification, DIRSYS displays the message:

NON-SYSTEM FILES(S) EXIST

The DIR command pauses after filling the screen. Press any key to continue the display.

Note: You can use the DEVICE command to change the number of columns displayed by DIR or DIRSYS.

Examples: A>DIR

Displays all DIR files catalogued in user 0 on the default drive A.

A>DIR B:

Displays all DIR files for user 0 on drive B.

A>DIR B:X.BAS

Displays the name X.BAS if the file X.BAS is present on drive B.

4A>DIR *.BAS

Displays all DIR files with filetype BAS for user 4 on drive A.

B>DIR A:X*.C?D

Displays all DIR files for user 0 on drive A whose filename begins with the letter X, and whose three-character filetype contains the first character C and last character D.

A>DIRSYS

Displays all files for user 0 on drive A that have the SYS attribute.

3A>DIRS *.CMD

This abbreviated form of the DIRSYS command displays all SYS files with filetype CMD on the default drive A for user 3.

Display Directory with Options

Syntax: DIR {d:} [options]
DIR {filespec} {filespec}...[options]

Explanation: The DIR command with options is an enhanced version of the DIR command. The DIR command displays CP/M-86 Plus files in a variety of ways. DIR can search for files on any or all drives, for any or all user numbers.

DIR allows the option list to occur anywhere in the command tail. These options modify the entire command line. Only one option list is allowed.

Options must be enclosed in left and right square brackets ("[" and "]"). The options can be used individually, or strung together separated by commas (",") or spaces. Options can be abbreviated to one or two letters if the abbreviation unambiguously identifies the option.

If a directory listing exceeds the size of your screen, DIR automatically halts the display when it fills the screen. Press any key to continue the display.

Table 6-3. DIR Display Options

Format: Option
Function

ATT
Displays the user-definable file attributes F1, F2, F3, and F4. See the SET utility for a description of file attributes.

DATE
Displays files with date and time stamps. If date and time stamping is not active, DIR displays the message:

    Date and Time Stamping Inactive.

DIR
Displays only files that have the DIR attribute.

DRIVE=ALL
Displays files on all accessed drives. DISK is also acceptable (in place of DRIVE) in all the DRIVE options.

DRIVE=(A,B,C,...,P)
Displays files on the drives specified.

DRIVE=d
Displays files on the drive specified by d.

EXCLUDE
Displays the files on the default drive and user area that do not match the files specified in the command line.

FF
Sends an initial Form-Feed (0Ch, or Ctrl-L) to the printer device if the printer is activated by Ctrl-P. If the LENGTH=n option is also specified, DIR issues a form-feed every n lines. Otherwise, the FF option deactivates the default paged output display.

FULL
Shows the name of the file, and the size of the file. The size is shown as the amount of space in kilobytes, and the number of 128-byte records allocated to the file. FULL also shows the attributes of the file. See the SET command for description of file attributes. If there is a directory label on the drive, DIR shows the password-protection mode and the time stamps. The display is alphabetically sorted. FULL is the default output format for display when using DIR with options.

LENGTH=n
Displays n lines of output before inserting a table heading; n must range from 5 to 65536. The default length is one full screen of information.

MESSAGE
Displays the names of the specified drives and user numbers it is currently searching.

NOPAGE
Continuously scrolls information by on the screen. Does not wait for you to press a key to restart the scrolling movement.

NOSORT
Displays files in the order it finds them on the disk. If this option is not included, DIR displays the files alphabetically.

RO
Displays only the files with the Read-Only attribute.

RW
Displays only the files that are set to Read/Write.

SIZE
Displays the filename and file size in kilobytes.

SYS
Displays only the files with the SYS attribute.

USER=ALL
Displays all files under all the user numbers for the default drive.

USER=n
Displays the files under the user number specified by n.

USER=(0,1,...,15)
Displays files under the user numbers specified.

Examples:

A>DIR C: [FULL]

The following is sample output of the [FULL] option display format:

Directory for Drive C: User 0

```
   Name     Bytes   Recs  Attributes  Prot      Update          Access
------------ ------ ------ ---------- ------ --------------- ---------------
TESTFILE BAK     1k      1 Dir RW      Read   09/01/02 13:04  09/01/02 13:07
TESTFILE TES     1k      1 Dir RO      None   09/01/02 13:07  09/01/02 13:09
TESTFILE Y       1k      1 Dir RW      None   08/25/02 03:33  08/25/02 03:33
TESTFILE ZZ      1k      1 Dir RW      None   08/25/02 03:36  08/25/02 03:36
SETDEF   CMD     4k     29 Dir RO      None                   08/25/02 03:36
SUBMIT   TX2     1k      1 Dir RO      None
SUBMIT   TX1     5k     43 Dir RO      None

Total Bytes     =     14k  Total Records =     77  Files Found =     7
Total 1k Blocks =     14   Used/Max Dir Entries for Drive C:   11/  64
```

A>DIR C: [SIZE]

The following is sample output of the [SIZE] option display format:

Directory for Drive C: User 0

```
C: TESTFILE BAK    1k : TESTFILE TES    1k : TESTFILE Y      1k
C: TESTFILE ZZ     1k : SETDEF   CMD    4k : SUBMIT   TX2    1k
C: SUBMIT   TX1    5k :
```

Total Bytes = 14k Total Records = 77 Files Found = 7
Total 1k Blocks = 14 Used/Max Dir Entries for Drive C: 11/ 64

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 disk, or on any drive with 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.

A>DIR [DRIVE=C,FF]

DIR sends a form-feed to the printer before displaying the files on drive C.

A>DIR D: [RW,SYS]

The preceding example displays all the files on drive D with Read/Write and SYS attributes.

A>DIR C: [USER=ALL]

Displays all the files under each user number (0-15) on drive C.

A>DIR [USER=2]

Displays all the files under user 2 on the default drive.

A>DIR C: [USER=(3,4,10)]

This example displays all the files under user numbers 3, 4, and 10 on drive C.

A>DIR [DRIVE=ALL]

Displays all the files under user 0 on all the drives in the drive search chain. (See the SETDEF command.)

4A>DIR [DRIVE=C]

Displays all the files under user 4 on drive C.

A>DIR [DRIVE=(B,D)]

Displays all the files under user 0 on drives B and D.

A>DIR [exclude] *.CMD

The preceding example lists all the files on the default drive and user 0 without a filetype of CMD.

A>DIR [user=all,drive=all,sys] *.PLI *.CMD *.A86

The preceding command line instructs DIR to list all the system files of type PLI, CMD, and A86 on the system in the currently active drives for all the user numbers on the drives.

A>DIR X.SUB [MESSAGE,USER=ALL,DRIVE=ALL]

The preceding command searches all drives under each user number for X.SUB. During the search, DIR displays the drives and user numbers.

A>DIR [drive=all user=all] TESTFILE.BOB

The preceding example instructs DIR to display the filename TESTFILE.BOB if it is found on any logged-in drive for any user number.

A>DIR [size,rw] D:

The preceding example instructs DIR to list each Read/Write file that resides on drive D with its size in kilobytes. Note that D: is equivalent to D:*.*.

The DUMP Command

Syntax: DUMP filespec

Explanation: Dump displays the contents of a file in hexadecimal and ASCII format.

Example: A>DUMP ASCII.BIN

Console output can look like the following:

```
CP/M-86 Plus -- DUMP Version 3.1
0000| 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  |................
0010| 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F  |................
0020| 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F  | !"#$%&'()*+,-./
0030| 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F  |0123456789:;<=>?
0040| 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F  |@ABCDEFGHIJKLMNO
0050| 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F  |PQRSTUVWXYZ[\]^_
0060| 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F  |`abcdefghijklmno
0070| 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F  |pqrstuvwxyz{|}~.
0080| 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F  |................
0090| 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F  |................
00A0| A0 A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF  |................
00B0| B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF  |................
00C0| C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF  |................
00D0| D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF  |................
00E0| E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF  |................
00F0| F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF  |................
```

The ED Command

Syntax: ED {input-filespec {d: | output-filespec}}

Explanation: The ED command lets you create and edit a disk file.

ED is a line-oriented context editor. You create and change character files line-by-line, or by individual characters within a line.

ED lets you create or alter the file named in the file specification. Refer to Section 7 for a more complete description of the ED command.

The ERASE Command

Syntax: ERASE {filespec} {[CONFIRM]} {[XFCB]}

Explanation: The ERASE command removes one or more files from a disk's directory in the current user number. Wildcard characters are accepted in the filespec. Directory and data space are automatically reclaimed for later use by another file. The ERASE command can be abbreviated to ERA.

Use the ERASE command with care, because all files in the current user number that satisfy the file specification are removed from the disk directory.

Command lines that take the form:

    ERASE {d:}wildcard-filespec

require your confirmation because they erase an entire group of files, not just one file. The system prompts with the following message:

    ERASE {d:}wildcard-filespec (Y/N)?

Respond with Y if you want to remove all matching files, and with N if you want to avoid erasing any files.

The CONFIRM option informs the system to prompt for verification before erasing each file that matches the filespec. You can abbreviate CONFIRM to C.

If you use the CONFIRM option with wildcard-filespec, then ERASE prompts for confirmation for each file. You can selectively erase the files you want by responding Y to the confirm message, or keep the files by responding N to the confirm message.

The XFCB option allows you to erase the additional directory elements used for password protection.

Examples: A>ERASE X.BAS

This command removes the file X.BAS from the disk in drive A.

    A>ERA *.PRN

The system asks to confirm:

```
ERASE *.PRN (Y/N)? Y
```

All files with the filetype PRN are removed from the disk in drive A.

```
B>ERA A:MY*.* [CONFIRM]
```

Each file on drive A with a filename that begins with MY is displayed with a question mark ("?") for confirmation. Type Y to erase the file displayed, N to keep the file.

```
A>ERA B:*.*
ERASE B:*.* (Y/N)? Y
```

All files on drive B are removed from the disk.

```
A>ERA A:L*.* [XFCB]
```

Every password directory entry for filenames beginning with L is erased from the directory of drive A.

The FORE Command

Syntax: FORE number

Explanation: The FORE command allocate CPU time to the foreground program.

Normally, the foreground program is allocated more time than a background program. The distributed system gives the foreground program 16 units of time and each background program 1 unit of time. The CCP built-in FORE command changes this ratio. For example, if you want to give the foreground 5 units of time and the background program 1 unit of time, enter the following:

Example: A>fore 5

The GET Command

Syntax: GET {CONSOLE INPUT FROM} FILE filespec {[{ECHO | NO ECHO} | SYSTEM]}
GET {CONSOLE INPUT FROM} CONSOLE

Explanation: The GET command directs CP/M-86 Plus to take console input from a file. The file can contain CP/M-86 Plus system commands and/or input for a user program. If you use the SYSTEM option, GET immediately takes the next system command from the file.

Console input is taken from a file until the program terminates, or until the file is exhausted. If the file is exhausted before program input is terminated, the program looks for subsequent input from the console. If the

program terminates before exhausting all its input, the system reverts back to the console for console input.

When the SYSTEM option is used, the system immediately goes to the file specified for console input. If you omit the SYSTEM option, you can enter one system command to initiate a user program whose console input is taken from the file specified in the GET command. The system reverts to the console for input when it reaches the end of the GET file input. The system also reverts to the console for console input if a GET CONSOLE INPUT FROM CONSOLE command (GET CONSOLE, in short) is included in the input file.

Get Console Input from a File

Syntax: GET {CONSOLE INPUT FROM} FILE filespec {[options]}

Explanation: The preceding form of the GET command tells the system to get subsequent console input from a file. The following table lists the GET options that you use in the following format:

[{ECHO | NO ECHO} | SYSTEM]

Table 6-4. GET Options

Format: Option
Meaning

ECHO
Specifies that the input is echoed to the console. This is the default option.

NO ECHO
Specifies that the file input is not to be echoed to the console. The program output and the system prompts are not affected by this option, and are still echoed to the console.

SYSTEM
Specifies that all system input is to be taken from the disk file specified in the command line. GET takes system and program input from the file until the file is exhausted, or until GET reads a GET CONSOLE command from the file.

Examples: A>GET FILE XINPUT
A>MYPROG

The preceding sequence of commands tells the system to activate the GET utility. However, because SYSTEM is not specified, the system reads the next input line from the console, and executes MYPROG. If MYPROG program requires console input, it is taken from the file XINPUT. When MYPROG terminates, the system reverts to the console for console input.

A>GET FILE XIN2 [SYSTEM]

The preceding command immediately directs the system to get subsequent console input from file XIN2, because it includes the SYSTEM option. The system

reverts to the console for console input when it reaches the end of file in XIN2. Or, XIN2 can redirect the system back to the console, if it contains a GET CONSOLE command.

Terminate Console Input from a File

Syntax: GET {CONSOLE INPUT FROM} CONSOLE

Explanation: The preceding form of the GET command tells the system to get console input from the console.

Example: A>GET CONSOLE

The preceding GET command tells the system to get console input from the console. You can use this command in a file (previously specified in a GET FILE command) that is already being read by the system for console input. It is used to redirect the console input to the console before the end of the file is reached.

The HELP Command

Syntax: HELP {topic}{subtopic1 subtopic2...subtopic8} {[NOPAGE | LIST]}
HELP [EXTRACT]
HELP [CREATE]

Explanation: The HELP command provides summarized information for all of the CP/M-86 Plus commands described in this manual. In the distributed CP/M-86 Plus system, HELP presents general information on a command as a topic, and detailed information on a command as a subtopic. HELP with no command tail displays a list of all the available topics. HELP with a topic in the command tail displays information about that topic, followed by any available subtopics. HELP with a topic and a subtopic displays information about the specific subtopic.

After HELP displays the information for your specified topic, it displays the special prompt HELP> on your screen. Subtopics can be accessed by preceding the subtopic with a period ("."). The period causes the subtopic search to begin at the last known level. You can continue to specify topics for additional information, or press ENTER to return to the CP/M-86 Plus system prompt.

You can abbreviate the names of topics and subtopics. Usually, one or two letters is enough to specifically identify the topics.

Display Information

Syntax: HELP topic {subtopic1...subtopic8} {[NOPAGE | LIST]}
HELP .subtopic

Explanation: The preceding forms of the HELP command display the information for the specified topic and subtopics. Use the following two options with this form of the HELP command:

- The NOPAGE option disables the default paged display of every n lines, where n is the number of lines per page as set by the system, or as set by the user with the SETDEF command. To stop the display, press Ctrl-S. To resume the display, press Ctrl-Q. You can abbreviate NOPAGE to N.

- The LIST option is the same as NOPAGE, except that it eliminates extra lines between headings. Use this option with Ctrl-P to list the help information on the printer.

Examples: A>HELP

HELP UTILITY V1.1

At "HELP>" enter topic {,subtopic}...

EXAMPLE: HELP> DIR EXAMPLES

Topics available:

COMMANDS CTRLCHARS BACK DATE DEVICE DIR
DUMP ED ERASE FDMAINT FILESPEC FORE
GET HDMAINT HELP INITDIR PATCH PIP
PUT RENAME SET SETDEF SHOW SUBMIT
TASK TYPE USER

HELP>

The preceding command displays a list of topics for which help is available...
(ROCHE> Except for a slight bug: the previous "EXAMPLE" is erroneous! It is now "DIR BUILT-IN EXAMPLES"..., because the EXAMPLES were separated in two categories, one for the built-in DIR command, and the other for the DIR with options command.)

A>HELP DATE

This command displays general information about the DATE command. It also displays any available subtopics.

A>HELP DIR OPTIONS [N]

The preceding command includes the subtopic OPTIONS. In response, HELP displays information about options associated with the DIR command. The display is not in paged mode.

A>HELP ED

The preceding command displays general information about the ED utility.

A>HELP ED COMMANDS

This form of HELP displays information about commands internal to ED. The preceding example can also be entered as

A>HELP ED
HELP>.COMMANDS

Add Your Own Descriptions to the HELP.HLP File

Syntax: HELP [EXTRACT]
HELP [CREATE]

Explanation: CP/M-86 Plus is distributed with two related HELP files: HELP.CMD and HELP.HLP. The HELP.CMD file is the command file that processes the text of the HELP.HLP file, and displays it on the screen. The HELP.HLP file is a text file to which you can add customized information, but you cannot directly edit the HELP.HLP file. You must use the HELP.CMD file to convert HELP.HLP to a file named HELP.DAT before you can edit or add your own text.

This form of the HELP command has the following options:

- The EXTRACT option accesses the file HELP.HLP on the default drive, and creates a file called HELP.DAT on the default drive. You can now invoke a word processing program to edit or add your own text to the HELP.DAT file. EXTRACT can be abbreviated to E.

- The CREATE option accesses your edited HELP.DAT file on the default drive, and builds a revised HELP.HLP file on the default drive. CREATE can be abbreviated to C.

You must add topics and subtopics to the HELP.DAT file in a specific format. A topic heading in the HELP.DAT file takes the form:

///nTopicname

The three backslashes are the topic delimiters, and must begin in column one. In the preceding format statement, n is a number from 1 through 9 that signifies the level of the topic. A main topic always has a level number of 1. The first subtopic has a level number of 2. The next level of subtopic has a level number of 3, and so forth, up to a maximum of nine levels. Topicname is the name of your topic, and allows a maximum of 12 characters and no spaces. Use an underline ("_") to represent a space. The entire line is terminated with a ENTER.

Use the following guidelines to edit and insert text into the HELP.DAT file.

- Place topics in alphabetical order.
- Place subtopics alphabetically within their respective main topic.
- Indicate levels by a number one through nine.

The following are some examples of topic and subtopic lines in the HELP.HLP

file:

```
///1NEW_UTILITY
///2COMMANDS
///3PARAMETERS
///2EXAMPLES
```

The first example illustrates the format of a main topic line. The second example shows how to number the first subtopic of that main topic. The third example shows how to number the next level subtopic under level 2. The fourth example shows how to return to the lower level subtopic. Any topic name with a level number of 1 is a main topic. Any topic name with a level number of 2 is a subtopic within its main topic.

When you are executing the HELP.CMD file, you need only enter enough letters of the topic to unambiguously identify the topic name. When referencing a subtopic, you must type the main topic name and the subtopic; otherwise, the HELP program cannot determine which main topic you are referencing. You can also enter a topic and subtopic following the program's internal prompt, HELP>:

```
HELP>ed commands
```

This form of HELP displays information about commands internal to the editing program, ED.

The INITDIR Command

Syntax: INITDIR d:

Explanation: The INITDIR command can modify a disk directory to allow date/time stamping of files on that disk, or remove date/time stamps.

You must use INITDIR to modify the directory for any disk on which you plan to record date/time stamps for your files. If the disk is blank, INITDIR modifies the directory to record date/time stamps. If files already exist on the disk, INITDIR checks the space available for date/time stamps in the directory. If there is not enough room for date/time stamps, INITDIR does not modify the directory, and returns an error message. However, because date/time stamps use 25% of your directory space, you cannot run INITDIR on a disk that has more than 75% of its directory in use. Use SHOW [DIR] to see how many free directory entries are left.

After you modify the directory for date/time stamps, you must use the SET command to specify time stamp options on the disk.

Example: A>INITDIR C:

The system prompts to confirm:

```
INITDIR WILL ACTIVATE TIME STAMPS FOR SPECIFIED DRIVE.
Do you really want to reformat the directory: C (Y/N)?
```

If the directory is modified for date/time stamps, INITDIR displays the message:

    Directory already reformatted
    Do you wish to recover date/time directory space (Y/N)?

Enter Y to modify the directory again, to eliminate date/time stamps. If you enter N, date/time stamping remains active on your disk, and INITDIR displays the following message:

    Do you want the existing date/time stamps cleared (Y/N)?

Enter Y to clear the existing stamps. Enter N to keep the existing date/time stamps.

The PATCH Command

Syntax: PATCH filespec {n}

Explanation: The PATCH command displays the patch numbers, or updates the patch history of a file, by including the patch number n in the CP/M-86 Plus system, or CP/M-86 Plus command, files.

Only CP/M-86 Plus files of filetype CMD or filetype SYS can be updated with the PATCH command.

The patch number n must be between 1 and 32, inclusive. To install a PATCH, follow the instructions supplied by Digital Research Inc.

Example: A>PATCH SHOW.CMD 2

The preceding command patches the system SHOW.CMD file with patch number 2. The system displays the following question:

    Do you want to indicate that Patch #2
    has been installed for SHOW.CMD? Y

If the patch is installed successfully, the system displays the message:

    Patch history installed

If the patch is not installed successfully, the system displays the following message:

    Patch history not installed

The PIP Command

Syntax: PIP dest-filespec | d:{[Gn]}=src-filespec{[o]}{,...} | d: {[o]}

Explanation: PIP copies one or more files from one disk and/or user number to another. PIP can rename a file after copying it. PIP can combine two or more files into one file. PIP can also copy a character file from the disk to the printer or to some other auxiliary, logical output device. PIP can create a file on a disk from input from the console, or from some other logical input device. PIP can transfer data from a logical input device to a logical output device, thus the name, Peripheral Interchange Program.

PIP copies file attributes with the file. This procedure includes Read/Write/Read-Only and SYS/DIR file attributes, and the user-definable attributes F1 through F4. If a file is password-protected, you must enter the password in the command line following the filename and/or filetype to which it belongs. If the password fails, the file is skipped, and the failure noted.

When you specify a destination file with a password, PIP assigns that password to the destination file, and automatically sets the password-protection mode to READ. You need a password to read the file. (See the SET command.) When you specify a destination file with no password, PIP does not assign a password to the destination file. When you specify only a destination drive, PIP assigns the same password and password-protection mode to the destination file as specified in the source file.

Single File Copy

Syntax: PIP d:{[Gn]}=src-filespec{[options]}
PIP dest-filespec{[Gn]}=d:{[options]}
PIP dest-filespec{[Gn]}=src-filespec{[o]}

Explanation: The first form shows the simplest way to copy a file. PIP looks for the file named by src-filespec (source filespec) on the default or optionally specified drive. PIP copies the file to the drive specified by d: and gives it the name specified by src-filespec. You can use the [Gn] option to place your destination file (dest-filespec) in the user number specified by n. The only option recognized for the destination file is [Gn]. Several options can be combined for the src-filespec. See Table 6-1.

The second form is a variation of the first. PIP looks for the file named by dest-filespec on the drive specified by d:, copies it to the default or optionally specified drive, and gives it the name specified by dest-filespec.

The third form shows how to rename the file after you copy it. You can copy it to the same drive and user number, or to a different drive and/or user number. Rules for options are the same. PIP looks for the file specified by src-filespec, copies it to the location specified in dest-filespec, and gives it the name indicated by dest-filespec.

Remember that PIP always goes to and gets from the current default user number, unless you specify otherwise with the [Gn] option.

Before you start PIP, you must have enough free space in kilobytes on your destination disk to hold the entire file or files that you are copying. Even

if you are replacing an old copy on the destination disk with a new copy, PIP still needs enough room for the new copy before it deletes the old copy. Use the DIR command to determine filesize, and the SHOW command to determine disk space. If there is not enough space, you can delete the old copy first by using the ERASE command.

Data is first copied to a temporary file, to ensure that the entire data file can be constructed in the space available on the disk. PIP gives the temporary file the filename specified for the destination, with the filetype $$$. If the copy operation is successful, PIP changes the temporary filetype $$$ to the filetype specified in the destination.

If the copy operation succeeds and a file with the same name as the destination file already exists, the old file with the same name is erased before renaming the temporary file.

File attributes (DIR, SYS, RO, RW) are transferred with the files.

If the existing destination file is set to Read-Only (RO), PIP asks you if you want to delete it. Answer Y or N. Use the [W] option to write over Read-Only files without verification.

You can include PIP options following each source name. There is one valid option ([Gn] -- go to user number n) for the destination file specification. Options are enclosed in left and right square brackets ("[" and "]"). Several options can be included for the source files. They can be packed together, or separated by spaces. Options can verify that a file is copied correctly, allow PIP to read a file with the system (SYS) attribute, cause PIP to write over Read-Only files, cause PIP to put a file into or copy it from a specified user number, transfer from lower- to uppercase, and much more.

Examples: A>PIP B:=A:oldfile.dat
A>PIP B:oldfile.dat=A:

Both forms of this command cause PIP to read the file OLDFILE.DAT from drive A and put an exact copy of it onto drive B. This is called the short form of PIP, because the destination (or source) names only a drive and does not include a filename. When using this form, you cannot copy a file from one drive and user number to the same drive and user number. You must put the destination file on a different drive or in a different user number. (See the section on PIP Options, and the USER Command.) The second short form produces exactly the same result as the first one. PIP looks for the file OLDFILE.DAT on drive A, the drive specified as the source.

A>PIP B:newfile.dat=A:oldfile.dat

This command copies the file OLDFILE.DAT from drive A to drive B, and renames it to NEWFILE.DAT. The file remains as OLDFILE.DAT on drive A. This is the long form of the PIP command, because it names a file on both sides of the command line.

A>PIP newfile.dat=oldfile.dat

Using this long form of PIP, you can copy a file from one drive and user

number (usually user 0, because CP/M-86 Plus automatically starts out in user 0 -- the default user number) to the same drive and user number. This long form gives you two copies of the same file on one drive and user number, each with a different name.

A>PIP B:PROGRAM.BAK=A:PROGRAM.DAT[G1]

The preceding command copies the file PROGRAM.DAT from user 1 on drive A to the current selected user number on drive B, and renames the filetype on drive B to BAK.

B>PIP program2.dat=A:program1.dat[E V G3]

In this command, PIP copies the file named PROGRAM1.DAT on drive A, and echoes [E] the transfer to the console, verifies [V] that the two copies are exactly the same, and gets [G3] the file PROGRAM1.DAT from user 3 on drive A. Because there is no drive specified for the destination, PIP automatically copies the file to the default user number and drive, in this case user 0 and drive B.

Multiple File Copy

Syntax: PIP d:{[Gn]}={d:}wildcard-filespec{[options]}

Explanation: When you use a wildcard in the source specification, PIP copies matching files one-by-one to the destination drive, retaining the original name of each file. PIP displays the message COPYING, followed by each filename as the copy operation proceeds. PIP issues an error message, and aborts the copy operation, if the destination drive and user number are the same as those specified in the source.

Examples: A>PIP B:=A:*.CMD

This command causes PIP to copy all the files on drive A with the filetype CMD to drive B.

A>PIP B:=A:*.*

This command causes PIP to copy all the files on drive A to drive B. You can use this command to make a back-up copy of your distribution disk. Note, however, that this command does not copy the CP/M-86 Plus system from the system tracks.

A>PIP B:=A:PROG????.*

The preceding command copies all files whose filenames begin with PROG from drive A to drive B.

A>PIP B:[G1]=A:*.BAS

This command causes PIP to copy all the files with a filetype of BAS on drive A in the default user number (user 0) to drive B in user number 1. Remember that the DIR, TYPE, ERASE, and other commands only access files in the same

user number from which they were invoked. (See the USER Command.)

Combining Files

Syntax: PIP dest-filespec{[Gn]}=src-filespec{[o]},src-filespec{[o]}{,...}

Explanation: This form of the PIP command lets you specify two or more files in the source. PIP copies the files specified in the source from left to right, and combines them into one file with the name indicated by the destination file specification. This procedure is called "file concatenation". You can use the [Gn] option after the destination file, to place it in the user number specified by n. You can specify one or more options for each source file.

Some of the options force PIP to copy files character-by-character. In these cases, PIP looks for a Ctrl-Z character to determine where the end of the file is. All of the PIP options force a character transfer, except the following:

A, C, Gn, K, O, R, V, and W.

Copying data to or from logical devices also forces a character transfer.

You can terminate PIP operations with Ctrl-C.

When concatenating files, PIP only searches the last record of a file for the Ctrl-Z end-of-file character. However, if PIP is doing a character transfer, it stops when it encounters a Ctrl-Z character.

Use the [O] option if you are concatenating machine code files. The [O] option causes PIP to ignore embedded Ctrl-Z (end-of-file) characters, which indicate the end-of-file character in text files, but might be valid data in object code files.

Examples: A>PIP NEWFILE=FILE1,FILE2,FILE3

The three files named FILE1, FILE2, and FILE3 are joined from left to right, and copied to NEWFILE.$$$. NEWFILE.$$$ is renamed to NEWFILE upon successful completion of the copy operation. All source and destination files are on the disk in the default drive A.

A>PIP B:X.BAS=Y.BAS, B:Z.BAS

The file Y.BAS on drive A is joined with Z.BAS from drive B, and placed in the temporary file X.$$$ on drive B. The file X.$$$ is renamed to X.BAS on drive B when PIP runs to successful completion.

Copy Files to and from Auxiliary Devices

Syntax: PIP dest-filespec {[Gn]}=src-filespec {[options]}
AUX: AUX: {[options]}

CON:	CON: {[options]}
LST:	NUL:
PRN:	EOF:

Explanation: This form is a special case of the PIP command line that lets you copy a file from a disk to a device, from a device to a disk, or from one device to another. The files must contain printable characters. Each peripheral device is assigned to a logical device. The logical device identifies a source device that can transmit data, or a destination device that can receive data. (See the DEVICE command.) A colon (":") follows each logical device name, so it cannot be confused with a filename. Enter Ctrl-C to abort a copy operation that uses a logical device in the source or destination.

The logical device names are listed as follows:

- CON: Console input or output device. When used as a source, the device is usually the keyboard; when used as a destination, it is usually the screen.

- AUX: Auxiliary Input or Output Device.

- LST: The destination device assigned to the list output device, usually the printer.

The following three device names have special meaning:

- NUL: A source device that produces 40 hexadecimal zeros.

- EOF: A source device that produces a single Ctrl-Z, the CP/M-86 Plus end-of-file mark.

- PRN: The printer device, with tab expansion to every 8th column, line numbers, and page ejects every 60th line.

Examples: B>PIP PRN:=CON:,MYDATA.DAT

Characters are first read from the console input device, usually the keyboard, and sent directly to your printer device. Ctrl-Z tells PIP that keyboard input is complete. At that time, PIP continues by reading character data from the file MYDATA.DAT on drive B. Because PRN: is the destination device, tabs are expanded, line numbers are added, and page ejects occur every 60 lines.

Note that, when the CON: device is the source, you must enter both the ENTER (producing a CR, Ctrl-M) and line-feed (LF, Ctrl-J) keys for a new line.

```
A>PIP B:FUNFILE.SUE=CON:
```

Whatever you type at the console is written to the file FUNFILE.SUE on drive B. End the keyboard input by typing a Ctrl-Z.

```
A>PIP LST:=CON:
```

Whatever you type at the console keyboard is written to the list device,

usually the printer. Terminate input with a Ctrl-Z.

```
A>PIP LST:=B:DRAFT.TXT[T8]
```

The file DRAFT.TXT on drive B is written to the printer device. Any tab characters are expanded to the nearest column that is a multiple of eight.

```
A>PIP PRN:=B:DRAFT.TXT
```

The preceding command causes PIP to write the file DRAFT.TXT to the list device. It automatically expands the tabs, adds line numbers, and ejects pages after 60 lines.

Copy Files to and from Special Devices INP: & OUT:

Experience has shown that, sooner or later, experienced users of CP/M-86 Plus are going to need to input data from, or output to, some kind of special devices. How to allow them to do that, without giving them the full source code of PIP? The solution is to provide them with the ability of "patching" themselves their own custom drivers inside PIP, provided that their code is less than 128 bytes long (in practice, this is very generous). Following is the source code of the user-customizable INP: and OUT: devices that are present in PIP.CMD, followed by a dump showing where they are in the ComManD file. Anybody understanding this section should be able to patch PIP without further explanations.

```
        NAME    inpout
;       CP/M-86 Plus Version 3.1 PIP Utility INP: / OUT:
;-------------------------------
        CSEG    0100h
;-------------------------------
; "Wrappers" for both special devices,
; so they work correctly under CP/M-86 Plus
; which is, let us not forget, a multi-tasking OS.

inpd:   push    bp
        call    inploc
        pop     bp
        ret

outd    push    bp
        mov     bp,sp
        mov     al,[bp]+4
        call    outloc
        pop     bp
        ret     2

; The 2 sample drivers are both 3-bytes long.

; The INP: example just returns an End-Of-File character in AL.

inploc  mov     al,01Ah
```

```
        ret

; The OUT: example just returns, without outputting AL.

outloc  ret
        nop
        nop

;--------------------------------
; To be sure that the record will
; be filled with 00h bytes.

        ORG     007Fh
        db      0
;--------------------------------
        END

A>dump pip.cmd
CP/M-86 Plus -- DUMP Version 3.1
0000| 01 BD 01 00 00 BD 01 00 00 02 79 00 00 00 79 02 |.........y...y.
0010| 00 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0020| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0030| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0040| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0050| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0060| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0070| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................

The Header Record of PIP.

0080| 9C 58 FA 8C D9 8E D1 8D 26 C0 06 50 9D E8 70 01 |.X......&..P..p.
0090| 33 C9 8B D1 CD E0 55 8B EC 8B 56 04 8B 4E 06 CD |3.....U...V..N..
00A0| E0 5D C2 04 00 00 00 00 00 00 00 00 00 00 00 00 |.]..............
00B0| 00 00 00 00 00 00 00 00 00 00 31 36 31 31 38 33 |..........161183
00C0| 43 50 2F 4D 2D 38 36 20 50 6C 75 73 00 00 00 00 |CP/M-86 Plus....
00D0| 43 4F 50 59 52 49 47 48 54 20 31 39 38 33 2C 20 |COPYRIGHT 1983,
00E0| 44 49 47 49 54 41 4C 20 52 45 53 45 41 52 43 48 |DIGITAL RESEARCH
00F0| 31 30 35 33 2D 30 36 37 33 2D 36 35 34 33 32 31 |1053-0673-654321
0100| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0110| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0120| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0130| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0140| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0150| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0160| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
0170| 00 43 53 45 47 20 70 61 74 63 68 20 61 72 65 61 |.CSEG patch area

The  standard beginning of a CP/M-86 Plus utility, with the date of  creation,
the  name  of  the Operating System, the copyright,  and  the  serial  number,
followed  by  the  patch area, in case PIP would need to  be  patched  in  the
future.

0180| 55 E8 0F 00 5D C3 55 8B EC 8A 46 04 E8 07 00 5D |U...].U...F....]
0190| C2 02 00 B0 1A C3 C3 90 90 00 00 00 00 00 00 00 |................
```

```
01A0| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
01B0| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
01C0| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
01D0| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
01E0| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
01F0| 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................
```

The record (128 bytes) holding the INP: & OUT: drivers. Using SID, you can check that the above bytes correspond to the source code of INPOUT.A86.

```
0200| 55 8B EC A0 32 01 F6 D0 D0 D8 73 6A B8 80 00 50 |U...2.....sj...P
0210| B8 E8 01 50 B0 80 50 E8 18 05 80 3E E8 01 00 B0 |...P..P....>....
0220| FF 74 01 40 A2 D8 01 E8 E5 02 3C 31 73 0A B8 CA |.t.@......<1s...
0230| 04 50 E8 B6 02 E8 60 02 B0 2D 50 B8 FF 00 50 E8 |.P....`..-P...P.
0240| 54 FE B0 6D 50 B8 01 00 50 E8 4A FE A0 D8 01 D0 |T..mP...P.J.....
0250| D8 73 0A B8 76 05 50 E8 81 02 E8 6D 02 E8 0A 04 |.s..v.P....m....
0260| A2 6A 02 A2 6B 02 E8 DE 03 A2 39 01 C6 06 CB 01 |.j..k.....9.....
0270| 01 C6 06 E5 01 00 80 3E E5 01 00 74 09 E8 CE 13 |.......>...t....
^C
A>That's all, folks!
```

Multiple Command Mode

Syntax: PIP

Explanation: This form of the PIP command starts the PIP utility, and lets you type multiple command lines while PIP remains in user memory.

PIP writes an asterisk ("*") on your screen when ready to accept input command lines.

You can type any valid command line described under previous PIP formats following the asterisk prompt.

Terminate PIP by pressing only ENTER following the asterisk prompt. The empty command line tells PIP to discontinue operation, and to return to the CP/M-86 Plus system prompt.

Examples:
```
A>PIP
    CP/M-86 Plus PIP Version 3.1
    *NEWFILE=FILE1,FILE2,FILE3
    *APROG.CMD=BPROG.CMD
    *A:=B:X.BAS
    *B:=*.*
    *ENTER      (=Press the ENTER key on your keyboard.)
    A>
```

This command loads the PIP program. The PIP command input prompt ("*") tells you that PIP is ready to accept commands. The effects of this sequence of commands are the same as in the previous examples, where the command line is included in the command tail. PIP is not loaded into memory for each command. To exit this PIP command mode, press ENTER or one of its equivalent control

characters, Ctrl-J or Ctrl-M, as shown.

Using Options with PIP

Explanation: With options, you can process your source file in many ways. You can expand tab characters, translate from upper- to lowercase, extract portions of your text, verify that the copy is correct, and much more.

The PIP options are listed in Table 6-5, using n to represent a number, and s to represent a sequence of characters terminated by a Ctrl-Z. An option must immediately follow the file or device it affects. The option must be enclosed in left and right square brackets ("[" and "]"). For those options that require a numeric value, no blanks can occur between the letter and the value.

You can include the [Gn] option after a destination file specification. You can include a list of options after a source file or source device. An option list is a sequence of single letters and numeric values that are optionally separated by blanks, and enclosed in left and right square brackets ("[" and "]").

Table 6-5. PIP Options

Option	Function
A	Copy only the files modified since the last copy. To back up only the files that have been modified since the last back-up, use PIP with the archive option, [A].
C	Prompt for confirmation before performing each copy operation. Use the [C] option when you want to copy only some files of a particular filetype.
Dn	Delete any characters past column n. This parameter follows a source file that 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 parameter follows a source name, PIP displays the source data at the console as the copy is taking place. The source must contain character data.
F	Filter Form-Feeds. When this parameter follows a source name, PIP removes all Form-Feeds (0Ch, or Ctrl-L) embedded 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 command to delete the old form-feeds, and a P command to simultaneously add new form-feeds to the destination file.
Gn	Get source from or go to user number n. When this parameter follows a source name, PIP searches the directory of user number n for the source file. When it follows the destination name, PIP places the destination file in the user number specified by n. The number must be

from 0 to 15.

H Hex data transfer. PIP checks all data for proper Intel hexadecimal file format. The console displays error messages when errors occur.

I Ignore :00 records in the transfer of Intel hexadecimal format file. The I option automatically sets the H option.

L Translate uppercase alphabetical letters in the source file to lowercase in the destination file. This parameter follows the source device or filename.

N Add line numbers to the destination file. When this parameter follows the source filename, PIP adds a line number to each line copied, starting with one, and incrementing by one. A colon (":") follows the line number. If N2 is specified, PIP adds leading zeros 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 machine code (non-character, hence non-printable) files. PIP ignores any Ctrl-Z/end-of-file during concatenation and transfer. Use this option if you are combining object code files.

Pn Set page length. n specifies the number of lines per page. When this parameter modifies a source file, PIP includes a page eject at the beginning of the destination file, and at every n lines. If n = 1 or is not specified, PIP inserts page ejects every 60 lines. When you also specify the F option, PIP ignores Form-Feeds (0Ch, or Ctrl-L) in the source data, and inserts new Form-Feeds in the destination data at the page length specified by n.

Qs Quit copying from the source device after the string s. When used with the S parameter, this parameter can extract a portion of a source file. The string argument must be terminated by Ctrl-Z.

Note: Because CP/M-86 Plus translates every character in a command line to uppercase, use this option in the PIP multiple command mode.

R Read system (SYS) files. Usually, PIP ignores files marked with the system attribute in the disk directory. But, when this parameter follows a source filename, PIP copies system files, including their attributes, to the destination.

Ss Start copying from the source device at the string s. The string argument must be terminated by Ctrl-Z. When used with the Q parameter, this parameter can extract a portion of a source file. Both start and quit strings are included in the destination file.

Note: Because CP/M-86 Plus translates every character in a command line to uppercase, use this option in the PIP multiple command mode.

Tn Expand tabs. When this parameter follows a source filename, PIP expands tab (Ctrl-I) characters in the destination file. PIP replaces

each Ctrl-I with enough spaces to position the next character in a column divisible by n.

U Translate lowercase alphabetical characters in the source file to uppercase in the destination file. This parameter follows the source device or filename.

V Verify that data is copied correctly. PIP compares the destination to the source data, to ensure that the data is written correctly. The destination must be a disk file.

W Write over files with RO (Read-Only) attribute. Usually, if a PIP command tail includes an existing RO file as a destination, PIP sends a query to the console to ensure that you want to write over the existing file. When this parameter follows a source name, PIP overwrites the RO file without a console exchange. If the command tail contains multiple source files, this parameter need follow only the last file in the list.

Z Zero the parity bit. When this parameter follows a source name, PIP sets the parity bit of each data byte in the destination file to zero. The source must contain character data.

Examples: A>PIP NEWPROG.BAS=CODE.BAS[L], DATA.BAS[U]

This command constructs the file NEWPROG.BAS on drive A by joining the two files CODE.BAS and DATA.BAS from drive A. During the copy operation, CODE.BAS is translated to lowercase, while DATA.BAS is translated to uppercase.

A>PIP CON:=WIDEFILE.BAS [D80]

This command writes the character file WIDEFILE.BAS from drive A to the console device, but deletes all characters following the 80th column position.

A>PIP B:=LETTER.TXT [E]

The file LETTER.TXT from drive A is copied to LETTER.TXT on drive B. The LETTER.TXT file is also written to the screen as the copy operation proceeds.

A>PIP LST:=B:LONGPAGE.TXT [FP65]

This command writes the file LONGPAGE.TXT from drive B to the printer device. As the file is written, form-feed characters are removed and reinserted at the beginning and every 65th line thereafter.

B>PIP LST:=PROGRAM.BAS [NT8U]

This command writes the file PROGRAM.BAS from drive B to the printer device. The N parameter tells PIP to number each line. The T8 parameter expands tabs to every eighth column. The U parameter translates lowercase letters to uppercase as the file is printed.

A>PIP
CP/M-86 Plus PIP Version 3.1

*PORTION.TXT=LETTER.TXT [SDear Sir^Z QSincerely^Z]

This command extracts a portion of the LETTER.TXT file from drive A by searching for the character sequence "Dear Sir" before starting the copy operation. When found, the characters are copied to PORTION.TXT on drive A until the sequence "Sincerely" is found in the source file. Because CP/M-86 Plus translates every letter in a command line to uppercase, this command only works when PIP is in the multiple command mode.

B>PIP B:=A:*.CMD [VWR]

This command copies all files with filetype CMD from drive A to drive B. The V parameter tells PIP to read the destination files, to ensure that data was correctly transferred. The W parameter lets PIP overwrite any destination files that are marked as RO (Read-Only). The R parameter tells PIP to read files from drive A that are marked with the SYS (System) attribute.

The PUT Command

Syntax: PUT {CONSOLE OUTPUT TO} FILE filespec {[options]}
PUT {PRINTER OUTPUT TO} FILE filespec {[options]}
PUT {CONSOLE OUTPUT TO} CONSOLE
PUT {PRINTER OUTPUT TO} PRINTER

Explanation: PUT allows you to direct the system to put console or printer output to a file for the next system command or user program entered at the console. Or, PUT directs all subsequent console or printer output to a file when you include the SYSTEM option.

Console output is directed to a file until the program terminates. Then, console output reverts to the console. Printer output is directed to a file until the program terminates. Then, printer output is directed back to the printer.

When you use the SYSTEM option, all subsequent console/printer output is directed to the specified file. This option terminates when you enter the PUT CONSOLE or PUT PRINTER command.

The option list has the syntax:

[{ECHO | NO ECHO} {FILTER | NO FILTER} | {SYSTEM}]

The PUT Options table defines the preceding option list.

Table 6-6. PUT Options

Format: Option
Meaning

ECHO
Output is echoed to the console. ECHO is the default option when you direct console output to a file.

NO ECHO
File output is not to be echoed to the console.

FILTER
Filtering of control characters is allowed, so control characters are translated to printable characters. For example, an escape character is translated to ^[.

NO FILTER
PUT does not translate control characters. This is the default option.

SYSTEM
System and program output are written to the file specified by filespec. Output is written to the file until a subsequent PUT CONSOLE command redirects console output back to the console.

Direct Console Output to a File

Syntax: PUT {CONSOLE OUTPUT TO} FILE filespec {[options]}

Explanation: The preceding form of the PUT command tells the system to direct subsequent console output to a file.

Example: A>PUT CONSOLE OUTPUT TO FILE XOUT [ECHO]

The preceding command directs console output to file XOUT, with the output echoed to the console.

Put Printer Output to a File

Syntax: PUT {PRINTER OUTPUT TO} FILE filespec {[options]}

Explanation: The preceding form of the PUT command directs printer output to a file.

The options are the same as in the PUT CONSOLE command, except that option NO ECHO is the default for the PUT PRINTER command. Note that, if ECHO is specified, printer output is echoed to the printer.

Examples: A>PUT PRINTER OUTPUT TO FILE XOUT
A>MYPROG

The preceding example directs the printer output of program MYPROG to file XOUT. The output is not echoed to the printer.

A>PUT PRINTER OUTPUT TO FILE XOUT2 [ECHO,SYSTEM]

The preceding command directs all printer output to file XOUT2 and to the printer, and the PUT is in effect until you enter a PUT PRINTER OUTPUT TO

PRINTER command.

The printer output can be directed to one or more files. The output to these files is terminated when you revert printer output to the printer using the following command:

```
A>PUT PRINTER OUTPUT TO PRINTER
```

Stop Console Output to a File

Syntax: PUT {CONSOLE OUTPUT TO} CONSOLE

Explanation: The preceding form of the PUT command directs console output to the console.

Example: A>PUT CONSOLE OUTPUT TO CONSOLE

The preceding command directs console output to the console.

Stop Printer Output to a File

Syntax: PUT {PRINTER OUTPUT TO} PRINTER

Explanation: The preceding form of the PUT command directs the printer output to the printer.

Example: A>PUT PRINTER OUTPUT TO PRINTER

The preceding example directs printer output to the printer.

The RENAME Command

Syntax: RENAME {new-filespec=old-filespec}

Explanation: The RENAME command lets you change the name of a file that is catalogued in the directory of a disk. It also lets you change several filenames if you use wildcards in the filespecs. You can abbreviate RENAME to REN.

New-filespec must not be the name of any existing file on the disk. Old-filespec identifies an existing file or files on the disk.

The RENAME command changes the file named by old-filespec to the name given as new-filespec.

RENAME does not make a copy of the file. RENAME changes only the name of the file.

If you omit the drive specifier, RENAME assumes that the file to rename is on the default drive. You can include a drive specifier as a part of the new name. If both file specifications name a drive, it must be the same drive.

If you use wildcards in the filespecs, the wildcards in the new filespec must correspond exactly to the wildcards in the old filespec. For example, in the following two commands, the wildcard filespecs correspond exactly:

```
A>REN *.TX1=*.TEX
A>REN A*.T*=S*.T*
```

In the following example, the wildcards do not match, and CP/M-86 Plus returns an error message.

```
A>REN A*.TEX=A.T*
```

Examples: A>RENAME NEWA86.BAS=OLDFILE.BAS

The file OLDFILE.BAS changes to NEWA86.BAS on drive A.

```
A>RENAME
```

The system prompts for the filespecs:

```
Enter New Name: X.PRN
Enter Old Name: Y.PRN
```

File Y.PRN is renamed X.PRN on drive A.

```
B>REN A:X.PAS=Y.PLI
```

The file Y.PLI changes to X.PAS on drive A.

```
A>RENAME S*.TEX=A*.TEX
```

The preceding command renames all the files matching the wildcard A*.TEX to files with filenames matching the wildcard S*.TEX, respectively.

```
A>REN B:NEWLIST=B:OLDLIST
```

The file OLDLIST changes to NEWLIST on drive B. Because the second drive specifier, B:, is implied by the first, it is unnecessary in this example. The preceding command line has the same effect as the following:

```
A>REN B:NEWLIST=OLDLIST
```

or

```
A>REN NEWLIST=B:OLDLIST
```

The SET Command

Syntax: SET [options]
SET d: [options]

SET filespec [options]

Explanation: The SET command initiates password protection and date/time stamping of files in the CP/M-86 Plus system. This command also sets file and drive attributes, such as the Read-Only, System, and user-definable attributes. You can label a disk, and protect the label with a password, with SET.

The SET command includes options that affect the disk directory, the drive, a file, or set of files. The discussion of the SET command explicitly states which of the three categories are affected.

To enable date/time stamping of files, you must first run INITDIR to format the disk directory.

Set File Attributes

Syntax: SET filespec [attribute-options]

Explanation: The preceding SET command sets the specified attributes of a file or a group of files.

Table 6-7. SET File Attributes

Format: Option
Meaning

DIR
Sets the file to the DIR attribute.

SYS
Gives the file the System (SYS) attribute.

RO
Sets the file attribute to allow Read-Only access.

RW
Sets the file attribute to allow Read/Write access.

ARCHIVE=OFF
Sets the archive attribute to OFF. The file is not backed up (archived). PIP with the [A] option can copy files with the archive attribute set to OFF. PIP with this option requires a wildcard filespec, and copies only files that were created or changed since the last time they were backed up with the PIP [A] option. PIP then sets the archive attribute to ON for each file successfully copied.

ARCHIVE=ON
Sets the archive attribute to ON. The file is backed up (archived). The archive attribute can be turned on explicitly by the SET command, or it can be turned on by PIP when copying a group of files with the PIP [A] option. The archive attribute is displayed by DIR.

F1,F2,F3,F4=ON | OFF
Turns on or off the file attribute F1, F2, F3, F4. Those bits are reserved for use by other Digital Research operating systems.

Examples: A>SET MYFILE.TEX [RO SYS]

The preceding command sets MYFILE.TEX to Read-Only and System.

A>SET MYFILE.TEX [RW DIR]

The preceding command sets MYFILE.TEX to Read/Write, with the Directory (DIR) attribute.

Set Drive Attribute

Syntax: SET {d:} [RO]
SET {d:} [RW]

Explanation: The preceding 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, ERASE cannot delete a file from it, and RENAME cannot rename a file on it. You cannot perform any operation that requires writing to the disk. When the specified drive is set to Read/Write, you can read or write to the disk in that drive. If you enter a Ctrl-C to the system prompt, all drives are reset to Read/Write.

Example: A>SET B: [RO]

The preceding command sets drive B to Read-Only.

Assign a Label to the Disk

Syntax: SET {d:} [NAME=labelname.typ]

Explanation: The preceding SET command assigns a label (name) to the disk in the specified or default drive.

CP/M-86 Plus provides a facility for creating a directory label for each disk. The directory label can be assigned an eight-character name and a three-character type, similar to a filename and filetype. Label names simplify cataloging disks and keeping track of different disk directories. The default label name is LABEL.

Example: A>SET [NAME=DISK100]

The preceding example labels the disk on the default drive DISK100.

Assign Password to the Label

Syntax: SET [PASSWORD=password]
SET [PASSWORD=ENTER (=Press the ENTER key on your keyboard.)

Explanation: The first form of the preceding SET command assigns a password to the disk label. The second form of the command removes password protection from the label.

You can assign a password to the label. If the label has no password, any user with access to the SET program can set other attributes to the disk, which might make the disk inaccessible to you. However, 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 if the label is password protected.

Examples: A>SET [PASSWORD=SECRET]
A>SET [PASSWORD=ENTER (=Press the ENTER key on your keyboard.)

The first command assigns SECRET to the disk label. The second command nullifies the existing password.

Note: If you use password protection on your disk, record the password. If you forget the password, you lose access to your disk or files.

Enable/Disable Password Protection for Files on Disk

Syntax: SET [PROTECT=ON]
SET [PROTECT=OFF]

Explanation: The first form of the SET command turns on password protection for all the files on the disk. The password protection must be on before you can assign passwords to individual files or commands.

The second SET command disables password protection for the files on your disk.

After a password is assigned to the label and the PROTECT option is on, you are ready to assign passwords to your files.

You can always determine if a disk is password-protected by using the SHOW command to display the label.

Assign Passwords to Files

Syntax: SET filespec [PASSWORD=password]

Explanation: The preceding SET command sets the password for filespec to the

password indicated in the command tail. Passwords can be up to eight characters long. Lowercase letters are translated to uppercase.

You can use wildcards in the filespec. SET assigns the specified password to the files that match the wildcard-filespec.

Note: Always record the passwords that you assign to your files. Without the password, you cannot access those files unless password protection is turned off for the whole disk. If you forget the password to the directory label, you cannot turn off the password protection for the disk.

Example: A>SET MYFILE.TEX [PASSWORD=ROSEBUD]

ROSEBUD is the password assigned to the file MYFILE.TEX.

Set Password-Protection Mode for Files with Passwords

Syntax: SET filespec [PROTECT=READ]
SET filespec [PROTECT=WRITE]
SET filespec [PROTECT=DELETE]
SET filespec [PROTECT=NONE]

Explanation: You can assign one of four modes of password protection to your file. The protection modes are READ, WRITE, DELETE, and NONE, and are described in the following table.

Table 6-8. Password-protection Modes

Format: Mode
Protection

READ
Required for reading, copying, writing, deleting, or renaming the file.

WRITE
Required for writing, deleting, or renaming the file. You do not need a password to read the file.

DELETE
Only required for deleting or renaming the file. You do not need a password to read or modify the file.

NONE
No password exists for the file. If a password exists, this modifier can be used to delete the password.

Example: B>SET *.TEX [PASSWORD=SECRET,PROTECT=WRITE]

The preceding command assigns the password SECRET to all the TEX files on drive B. Each TEX file is given a WRITE protect mode, to prevent unauthorized editing.

Assign a Default Password

Syntax: SET [DEFAULT=password]

Explanation: The preceding SET command assigns a default password for the system to use during your computer session. The system uses the default password to access password-protected files if you do not specify a password, or if you enter an incorrect password. The system lets you access the file if the default password matches the password assigned to the file.

Example: A>SET [DEFAULT=dd]

The preceding command instructs the system to use DD as a password if you do not enter a password for a password-protected file.

Set Date/Time Stamp Options on Disk

Syntax: SET [CREATE=ON]
SET [ACCESS=ON]
SET [UPDATE=ON]

Explanation: The preceding SET commands allow you to keep a record of the date and time of file creation and update, or of the last access and update of your files.

[CREATE=ON]
Turns on CREATE date/time stamps on the disk in the default drive. To record the creation date and time of a file, the CREATE option must have been turned ON before the file is created.

[ACCESS=ON]
Turns on ACCESS date/time stamps on the disk in the default drive. ACCESS and CREATE options are mutually exclusive. Only one can be in effect at a time. If you turn ON the ACCESS date/time stamp on a disk that has the CREATE date/time stamp, the CREATE date/time stamp is automatically turned OFF.

[UPDATE=ON]
Turns on UPDATE date/time stamps on the disk in the default drive. UPDATE date/time stamps record the last time the file is modified.

To enable date/time stamping, you must first run INITDIR to format the disk directory for time and date stamping.

Although there are three kinds of date/time stamps, only two stamps can be given to the files on a single disk. The UPDATE date/time stamp can be used with either the CREATE stamp or the ACCESS stamp, but you cannot use the ACCESS stamp with the CREATE date/time stamp.

Note that most text editors make a copy of the file to be edited, then replace

the old version of the file with the edited version at the end of the edit session. The edited file (the new version) is treated as a new file. The original CREATE date/time stamp is changed to the date and time that the file is edited, making the UPDATE and CREATE date/time stamps the same. For this reason, assigning both CREATE and UPDATE to files that are edited with a text editor is redundant. This combination of stamps works well with data files to be updated by another program that writes, or appends data, to the file.

Examples: A>SET [ACCESS=ON]

The DIR with [FULL] option displays the following date and time stamps:

```
B>DIR [FULL]

Directory for Drive B:

   Name      Bytes  Recs  Attributes  Prot      Update         Access
------------ -----  ----  ----------  ---- -------------- --------------
ONE     .TEX     9k    71  Dir RW      None             08/03/01 10:56
THREE   .TEX    12k    95  Dir RW      None               08/05/01 15:45
TWO     .TEX    10k    76  Dir RW      None             08/10/01 09:13
```

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

```
A>SET [CREATE=ON,UPDATE=ON]
```

The following DIR output shows how files with CREATE and UPDATE date/time stamps are displayed.

```
B>DIR [FULL]

Directory for Drive B:

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

Additional SET Examples

Examples: A>SET *.CMD [SYS,RO,PASS=123,PROT=READ]

The preceding setting gives the most protection for all the CMD files on drive A. With the password-protection mode set to READ, you cannot even read one of the CMD files without entering the password 123, unless the default password is set to 123. Even if the correct password is entered, you cannot write to the file, because the file is Read-Only.

```
A>SET *.CMD [RW,PROTECT=NONE,DIR]
```

The preceding command reverses the protection and access attributes of the CMD files affected by the previous example. After executing the preceding command, there is no password protection, the files of type CMD can be read from or written to, and are set to DIR files.

The SETDEF Command

Syntax: SETDEF {d:{,d:{,d:{,d:}}}} {[TEMPORARY=d:] | [ORDER=(typ {,typ})]}
SETDEF [DISPLAY | NODISPLAY]
SETDEF [PAGE | NOPAGE]

Explanation: The SETDEF command lets you display or define the disk search order, the temporary drive, and the filetype search order. The SETDEF definitions affect only the loading of programs (CMD) and/or execution of SUBMIT (SUB) files. The SETDEF command also lets you turn on/off the DISPLAY and PAGE modes for the system. When DISPLAY mode is on, the system displays the location and name of programs loaded, or SUB files executed. When PAGE mode is ON, CP/M-86 Plus utilities stop after displaying one full screen of information. Press any key to continue the display.

The system usually searches the specified drive or the default drive for files. The user can use the SETDEF command, to extend the search for program files and submit files, for execution purposes only.

Note: A CP/M-86 Plus program file has a filetype of CMD. A file containing commands to be executed by SUBMIT has a filetype of SUB.

Display the Program Loading Search Definitions

Syntax: SETDEF

Example: A>SETDEF

```
Drive Search Path:
1st Drive          - Default

Search Order       - CMD
Temporary Drive    - Default
Console Page Mode  - Off
Program Name Display - Off
```

Explanation: The preceding form of the SETDEF command displays the disk search order, the temporary drive, and the filetype search order.

Assign the Drive for Temporary Files

Syntax: SETDEF [TEMPORARY=d:]

Explanation: The preceding form of the SETDEF command defines the disk drive to be used for temporary files. The default drive used for temporary files is the system default drive.

Example: A>SETDEF [TEMPORARY=M:]

The preceding command sets disk drive M as the drive to be used for temporary files.

Define the Disk Drive Search Order

Syntax: SETDEF {d: {,d: {,d: {,d:}}}}

Explanation: The preceding form of the SETDEF command defines the disks to be searched by the system for programs and/or submit files to be executed. The CP/M-86 Plus default is to search only the default drive.

Note: @ can be substituted for d: to indicate that the default drive is to be included in the drive search order.

Example: A>SETDEF C:,@

The preceding example tells the system to search for a program on drive C, then, if not found, search for it on the default drive.

Define the Filetype Search Order

Syntax: SETDEF [ORDER = (typ {,typ})]

where typ = CMD or SUB

Explanation: The preceding form of the SETDEF command defines the filetype search order to be used by system for program loading. The filetype, indicated as typ in the syntax line, must be CMD or SUB. The CP/M-86 Plus default search is for CMD files only.

Example: A>SETDEF [ORDER=(CMD,SUB)]

The preceding command instructs the system to search for a CMD file to execute. If no CMD file is found, search for a SUB file.

Turn On/Off System Display Mode

Syntax: SETDEF [DISPLAY | NODISPLAY]

Explanation: The preceding command turns the system display mode ON or OFF. The default system display mode is OFF. When the display mode is ON, CP/M-86

Plus displays the following information about a program file before loading it for execution: drive, filename, filetype (if any), and user number (if not the default user number).

Example: A>SETDEF [DISPLAY]

The preceding command turns ON the system display mode. The system now displays the name and location of programs loaded, or submit files executed. For example, if you enter the PIP command after turning ON the system display mode, CP/M-86 Plus displays the following:

```
A>PIP
A:PIP      CMD
CP/M-86 Plus PIP Version 3.1
*
```

indicating that the file PIP.CMD was loaded from drive A under the current user number. If the current user number is not 0, and if PIP.CMD does not exist under the current user number, then the system displays the location of PIP.CMD as follows:

```
4A>PIP
A:PIP      CMD  (User 0)
CP/M-86 Plus PIP Version 3.1
*
```

indicating that PIP.CMD was loaded from drive A under user number 0. This mode is in effect until you enter the following command to turn off the system DISPLAY mode:

```
SETDEF [NODISPLAY]
```

Turn On/Off System Page Mode

Syntax: SETDEF [PAGE | NOPAGE]

Explanation: The preceding command turns on/off the system page mode. When the PAGE mode is set to on, CP/M-86 Plus utilities stop after displaying one full screen of information, called a console page. The utilities resume after you press any key.

The default setting of the system page mode is ON.

Example: A>SETDEF [NOPAGE]

The preceding command turns OFF the system page mode. CP/M-86 Plus utilities do not pause after displaying a full console page, but continue to scroll.

The SHOW Command

Syntax: SHOW {d:} {[SPACE | LABEL | USERS | DIR | DRIVE | PROGRAM]}

Explanation: The SHOW command displays the following disk drive information:

- access mode, and amount of free disk space
- disk label
- current user number
- number of files for each user number on the disk
- number of free directory entries for the disk
- drive characteristics
- all background programs

Display Access Mode and Disk Space Available

Syntax: SHOW {d:} {[SPACE]}

Explanation: The preceding form of the SHOW command displays the drive, the access mode for that drive, and the remaining space in kilobytes for the specified drive. SHOW alone displays the information for all logged-in drives in the system.

Examples:

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

A>SHOW
A: RW,  Space:      4k
B: RW,  Space:   9,488k
```

The first example shows that drive B has Read/Write access and has 9,488K bytes of space left. The second example shows that drive A also is Read/Write and has only 4K bytes left, and drive B is Read/Write and has 9,488K bytes left.

Display Disk Label

Syntax: SHOW {d:} [LABEL]

Explanation: The preceding form of the SHOW command displays disk label information.

Example: A>SHOW B: [LABEL]

The preceding command displays the following for drive B:

Label for drive B:

```
Directory  Passwds   Stamp  Stamp  Label Created  Label Updated
Label      Reqd      Create Update
```

---------- ------- ------ ------ ------------- -------------
TOMSDISK   on   on   on   07/04/83 10:30 07/08/03 09:30

The first column, Directory Label, displays the name assigned to that drive directory. The second column, Passwds Reqd, shows that password protection is turned ON for that drive.

As described in the SET command, 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 date and time of the last access to the file. Access is defined as reading from or writing to the file. The third column of the SHOW LABEL output displays both the type of stamp and if it is ON. In the preceding example, creation time stamps are given to new files, as shown by the "Stamp Create" column heading.

The fourth column displays the status of the second time stamp field, the update time stamp. Update time stamps display the date and time of the last update to a file, that is to say, the last time someone wrote to the file. In the SHOW LABEL display, update time stamps are ON.

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

Display User Number Information

Syntax: SHOW {d:} [USERS]

Explanation: The preceding command displays the current user number and all the user numbers used on the drive, and the corresponding number of files assigned to them.

Example:

```
A>SHOW [USERS]
Active User :    1
Active Files:    0   2   3   4
A: # of files:  95  40   1  26
A: Number of free directory entries: 350
```

Display Number of Free Directory Entries

Syntax: SHOW {d:} [DIR]

Explanation: The preceding command displays the number of free directory entries on the specified drive.

Example:

```
A>SHOW C: [DIR]
```

C: Number of free directory entries: 24

The preceding command shows that there are only 24 free directory entries on drive C.

Display Drive Characteristics

Syntax: SHOW {d:} [DRIVE]

Explanation: The preceding form of the SHOW command displays the drive characteristics of the specified drive.

Example: A>SHOW [DRIVE]

The following is an example of the system display for the preceding command:

```
    A: Drive Characteristics
5,616: 128 Byte Record Capacity
  702: Kilobyte Drive  Capacity
  256: 32 Byte  Directory Entries
  256: Checked  Directory Entries
  128: Records / Directory Entry
   16: Records / Block
    9: Sectors / Track
    4: Reserved  Tracks
  512: Bytes / Physical Record
```

Display Currently Active Processes

Syntax: SHOW [PROGRAM]

Explanation: The preceding form of the SHOW command displays all the currently running programs by program name and program ID. This information can be used by TASK to terminate a running program.

Example:

```
C>back pip lst:=temp.txt [ofile=temp1]

Background output in file: TEMP1
C>back pip lst:=temp.txt [ofile=temp2]

Background output in file: TEMP2
C>back pip lst:=temp.txt [ofile=temp3]

Background output in file: TEMP3
C>

C>show [program]
0:  PIP
```

1: PIP
2: PIP
3: SHOW

The SORT Command

Syntax: SORT {CFILE=filename}

Explanation: The SORT command lets you sort or merge one or more (up to 32) files, following several criteria. It is to data files what WordStar is to text files.

SORT is the only non-Digital Research utility provided with CP/M-86 Plus. It is so powerful that 100 pages are needed to document all the things that it can do... As a result, it has its own separate manual, that you are counselled to print, to always have it available near your console.

Following is a summary of what SORT can do on your CP/M-86 Plus computer:

Sort/Merge operation
---

* SORTS up to 32 input files into a single output file, automatically using external merge as necessary, depending on amount of data and amount of memory.

* MERGES up to 32 pre-sorted files, all of which (memory permitting) are read in parallel for efficient true merge operation.

* Sort and merge input files can be specified IN THE SAME RUN. Thus, sorting new detail records, and integrating them into an already-ordered master file, is a simple and efficient SORT operation.

* Record selection, file conversions, and other features can be used, independent of the sort/merge process (by specifying a single input merge).

Flexible file and record formats
---

* Handles diskette files compatible with BASIC, FORTRAN (formatted or binary I/O), COBOL, Assembler, and text editors.

* Handles files up to the file size limit imposed by the operating system.

* Handles up to 65K logical records per file.

* Files may contain ASCII, BCD, and/or BINARY data.

* Handles LOGICAL RECORD LENGTHS to 4096 characters.

* Files may contain FIXED LENGTH records, varying-length CARRIAGE-RETURN-DELIMITED records, or COBOL-style VARIABLE-LENGTH record with length at beginning; COBOL RELATIVE FILES are also supported.

* FIELDS used as sort keys, or in record selection tests, may have fixed COLUMN-SPECIFIED position and length, or variable COMMA-DELIMITED position and length.

Multiple sort keys with many options and data types
---

* One to 32 key fields may be specified, each with an independent ASCENDING/DESCENDING indicator, collating sequence options, and data type attributes.

* Key data may be ASCII STRING text, ASCII NUMERIC, BCD (COBOL packed decimal), or BINARY.

* NUMERIC-ASCII option sorts on free-format numbers (including exponential notation) as typically output by the PRINT statement of BASIC, or D, E, F, or G format in FORTRAN.

* BINARY data types handled include FIXED POINT of any length, signed or unsigned, stored low-high or high-low, and MicroSoft FLOATING POINT, single or double precision. This includes the INTEGER, SINGLE PRECISION (REAL), and DOUBLE PRECISION data types of MicroSoft MBASIC and FORTRAN.

* User-command-specified ALTERNATE COLLATING SEQUENCE and EBCDIC collating sequence provided on an individual field basis.

* Additional options include treating lower case as upper case, treating last rather than first character of key as most significant, and ignoring the high-order (parity) bit.

Powerful record selection
---

* Desired records can be extracted from a file via SELECT and EXCLUDE commands which act on both sort and merge-only inputs.

* Selection is specified via ANY NUMBER OF CONDITIONAL TESTS of a field against a specified value or range of fixed values, or against another field in the same record.

* VALUES to test against are usually specified as TEXT STRINGS, but may also be specified as BCD numbers, or as binary values expressed in OCTAL, DECIMAL, or HEXADECIMAL.

* Test OPERATORS include the usual LESS THAN, EQUAL TO, etc, and also

BETWEEN and NOT BETWEEN. Tests may optionally be combined with AND, OR, and EXCLUSIVE OR operations.

* Record selection tests can use fixed position or comma-delimited fields, and all the data types and attributes specified above for keys: ASCII, binary, numeric-ASCII, BCD, Floating Point, alternate collating sequences, etc.

Additional features

* EXTRACTION BY RECORD NUMBER: Starting and/or ending record numbers may be specified for each sort input file.

* OUTPUT DISKETTE CHANGE option makes it possible, in most cases, to sort an entire diskette of data in a two-drive system without writing over the input file.

* TAGSORT option reduces work file diskette space requirements by using pointers to records, rather than full records, during sorting and merging, then retrieving the input records via a high-speed random access algorithm while writing the output file.

* CONSOLE PRINTOUTS: user can select five degrees of message printout including none, a brief message giving number of records processed, or a detailed breakdown showing numbers of records input, sorted, merged, output, inserted, deleted, etc. Printout of disk space usage may also be requested.

* UTILITY FUNCTIONS: SORT can perform many useful functions, such as converting files to a different record type, changing record lengths, converting records with comma-delimited fields to fixed-position fields, and extracting or re-arranging (positional) fields within the record. Data may be simultaneously sorted, or kept in input order.

Optional outputs

* KEYS-ONLY OUTPUT: With this option, the output file receives the specified KEYS as extracted from the input records, rather than the sorted input records. This permits building an index to a file, extracting fields in order to print a summary, or selecting and re-arranging (positional) fields to form a new data base.

* RECORD NUMBER OUTPUT: The output file receives the record numbers of the input records, in either ASCII or binary format. The extracted keys may accompany the record numbers, or not. This permits building MULTIPLE INDICES into a file, ordered on various keys, without duplicating the data, so that another program may retrieve the records by key value or in key order.

* POINTER OUTPUT: The output file receives the sector number and byte

offset at which each record begins, with or without the keys. This provides another method of building indices whereby another program may retrieve the records from the original file.

Compatible with BASIC
---

* Handles CARRIAGE-RETURN-DELIMITED RECORDS of varying lengths.

* Handles COMMA-DELIMITED FIELDS, with or without spaces surrounding the data. QUOTES may be used to embed commas, or leading or trailing blanks.

* FREE FORMAT NUMBERS can be used as keys, or in record selection tests. SORT correctly interprets the value of variable-length signed or unsigned numbers with varying point position (or omitted decimal point), and the optional use of exponentional (E) notation.

* It is un-necessary to use special programming techniques to produce fixed record lengths, or field positions, or to align decimal points in numeric data.

* Variable-length records and variable-length fields can REDUCE DISK FILE SIZES, whether or not the file is to be processed by BASIC.

* Also handles binary INTEGER, SINGLE PRECISION, and DOUBLE PRECISION data.

Easy operation
---

* INTERACTIVE KEYWORD COMMAND INPUT for easy operator entry of sort/merge specifications, and easy error correction.

* COMMAND FILES may be used to simplify operator invocation of regularly-used SORT procedures.

* ERROR MESSAGES contain explicit and understandable text.

High performance
---

* Uses fast HEAPSORT algorithm with "sort-stretch" modification, dynamic merge optimization, and I/O buffering strategies designed to maximize throughput in a floppy disk environment.

* ADJUSTS its internal memory allocation, I/O buffer sizes, and other variables in response to various numbers of inputs, record lengths, options invoked, and amount of RAM memory available. Usable in a minimal system; runs faster and uses less disk work space in systems with additional RAM.

The SUBMIT Command

Syntax: SUBMIT {filespec} {argument} ... {argument}

Explanation: The SUBMIT command lets you execute a group or batch of commands from a SUB file, which is a file with filetype of SUB.

Usually, you enter commands one line at a time. If you must enter the same sequence of commands several times, you might find it easier to batch the commands together, using the SUBMIT command. To do this, create a file and enter your commands in this file. The file is identified by the filename, and must have a filetype of SUB. When you issue the SUBMIT command, SUBMIT reads the file named by the filespec, and prepares it for interpretation by CP/M-86 Plus. When the preparation is complete, SUBMIT sends the file to CP/M-86 Plus line-by-line, as if you were typing each command.

You create the SUB file with a text edition. It can contain CP/M-86 Plus commands, nested SUBMIT commands, and input data for a CP/M-86 Plus command or a program. CP/M-86 Plus command lines cannot exceed 128 characters.

You can pass arguments to SUB files when you execute them. Each argument you enter is assigned to a parameter in the SUB file. The first argument replaces every occurrence of $1 in the file, the second argument replaces parameter $2, and so on, up to parameter $9. For example, if your file START.SUB contains the following commands:

```
ERA $1.BAK
DIR $1
PIP $1=A:$2.CMD
```

and you enter the following SUBMIT command:

```
A>SUBMIT START SAM TEX
```

The argument SAM is substituted for every $1 in the START.SUB file, and TEX for every occurrence of $2 in the START.SUB file. SUBMIT then creates a file with the parameter substitutions, and executes this file. This file now contains the following commands:

```
ERA SAM.BAK
DIR SAM
PIP SAM=A:TEX.CMD
```

If you enter fewer arguments in the SUBMIT command than parameters in the SUB file, the remaining parameters are not included in the commands.

If you enter more arguments in the SUBMIT command than parameters in the SUB file, the remaining arguments are ignored.

To include an actual dollar sign ("$") in your SUB file, type two dollar signs ("$$"). SUBMIT replaces them with a single dollar sign when it substitutes an

argument for a parameter in the SUB file. For example, if file AA.SUB contains line:

```
ASM $1 $$$2
```

and you enter the following SUBMIT command:

```
A>SUBMIT AA FILE PZ
```

then the translated file contains the following:

```
ASM FILE $PZ
```

Program Input Lines in a SUB File

A SUB file can contain program input lines. Any program input is preceded by a less than sign ("<") as in the following example:

```
PIP
<B:=*.A86
<CON:=DUMP.A86
<
DIR
```

The three lines after PIP are input lines to the PIP command. The third line consists only of the less than sign ("<"), indicating a ENTER. The ENTER causes PIP to return to the system, to execute the final DIR command.

If the program terminates before using all of the input, SUBMIT ignores the excess input lines, and displays a warning message.

If the program requires more input than is in the SUB file, it expects you to enter the remaining input from the keyboard.

You can enter control characters in a SUB file by using the usual convention of preceding the control character by an up-arrow character [^] followed by the letter to be converted to a control character. To enter an actual ^ character, use the combination ^^. This combination translates to a single ^ in the same manner that $$ translates to a single $.

The SUB File

The SUB file can contain the following types of lines:

- any valid CP/M-86 Plus command
- any valid CP/M-86 Plus command with SUBMIT parameters ($0 to $9)
- any data input line
- any program input line with SUBMIT parameters ($0 to $9)

Some applications take advantage of an additional SUBMIT feature. If a command

line begins with a colon (":"), it is not executed unless the previous command was successfully executed. This is useful in cases where one command utilizes the results of another, as in the following example:

```
A>SUBMIT SUBFILE
A>COMPUTE RESULTS.DAT
A>:REPORT RESULTS.DAT
```

Example: The following lines illustrate the variety of lines that can be entered in a SUB file:

```
DIR
DIR *.BAK
ASM $1 $$$4
PIP PRN:=$1.LST[T$2 $3 $5]
DIR *.A86
PIP
<B:=*.A86
<CON:=DUMP.A86
<
DIR B:
```

Executing the SUBMIT Command

Syntax: SUBMIT

```
SUBMIT filespec
SUBMIT filespec argument ... argument
```

If you enter only SUBMIT, the system prompts for the rest of the command. You enter the filespec and arguments.

Example: A>SUBMIT

The system displays the following prompt. Enter filespec and arguments here, such as:

```
Enter File to Submit:  START B TEX
```

Another example might be

```
A>SUBMIT SUBA
```

Still another example, using parameters, is

```
A>SUBMIT AA ZZ SZ
```

where AA is the SUB file AA.SUB, ZZ is the argument to replace any occurrences of $1 in the AA.SUB file, and SZ is the argument to replace all occurrences of $2 in the AA.SUB file.

The STARTUP.SUB Start-up File

Every time you turn on or reset your computer, CP/M-86 Plus automatically looks for a special SUB file named STARTUP.SUB to execute. If it does not exist, then CP/M-86 Plus resumes normal operation. If the STARTUP.SUB file exists, the system executes the commands in the file. This file is convenient to use if you regularly execute a set of commands before you do your regular session on the computer. For example, if you always want to enter the current date and time on your computer before you enter any other commands, you can create the STARTUP.SUB file, with ED, and enter the DATE command as follows:

```
DATE SET
```

Then, whenever you bring up the system, the system executes the DATE command, and prompts you to enter the date and time. By using this facility, you can be sure to execute a regular sequence of commands before starting your usual session.

$GLOBAL and $LOCAL Variables

There are two commands that you can use in SUBMIT files that can alter processing at the system level after the SUBMIT file finishes executing. You can write a file that resets the user number with the USER utility, or changes the default disk with the d: convention.

If you only want these changes to affect program execution within the SUBMIT file, preface these commands with the $LOCAL option. When the SUBMIT file is finished executing, the user number and/or drive number return to their former values. This is the default operation of SUBMIT.

If you want these changes to remain valid after the execution of the SUBMIT file, preface these commands with the $GLOBAL option. When the SUBMIT file is finished executing, the user number and/or drive number retain the values specified in the SUBMIT file.

Example: The following program limits the action of the drive specification and the USER command to within the SUBMIT file. When the SUBMIT file is finished executing, the user number and the drive number return to their former values.

```
$LOCAL
B:
USER 2
PAYROLL.CMD
```

The following program lets the action of the drive specification and the USER command remain after the end of the SUBMIT file execution. When the SUBMIT file is finished executing, the user number and drive number retain the values specified in the SUBMIT file. A program like this is useful as a System file located in user 0 in the system drive.

```
$GLOBAL
```

```
B:
USER 2
DIR
```

The $GLOBAL option is ignored if the SUBMIT program is running in the background.

The TASK Command

Syntax: TASK {{[program ID]} {[program name]}}

Explanation: TASK allows you to stop (terminate) a program running in the background, or to display all programs running in the background. TASK with no command tail displays the current program running in the background, and prompts you for which one to terminate. If you do not want to stop any of the programs, press ENTER to return to the CCP.

Either the program ID or the program name can be specified. However, only the program ID is guaranteed to specify the program. If the program name is specified and it does not result in uniquely identifying the program, then TASK displays all programs with the same name, and asks you to select which program is to be terminated. Again, a ENTER does nothing but returns you back to the CCP. See the discussion on background tasks in Section 5.

Examples:

```
C>back pip lst:=temp.txt [ofile=temp1]

Background output in file: TEMP1
C>back pip lst:=temp.txt [ofile=temp2]

Background output in file: TEMP2
C>back pip lst:=temp.txt [ofile=temp3]

Background output in file: TEMP3
C>

C>task
0:  PIP
1:  PIP
2:  PIP
3:  TASK
Stop which program?

A>TASK 0
```

Stops background program number 0.

```
A>TASK RASM86
```

Stops RASM86.CMD that is running in the background.

A>TASK

Displays the current background programs, and asks the user which one to terminate.

The TYPE Command

Syntax: TYPE {filespec {[PAGE]} | [NOPAGE]}

Explanation: The TYPE command displays the contents of one or several ASCII character file(s) on your screen. The PAGE option displays the console listing in paged mode, which means that the console listing stops automatically after listing n lines of text. (See the DEVICE command to set n to a different value.) Press any character to continue listing another n lines of text. Press Ctrl-C to exit back to the system. PAGE is the default mode.

The NOPAGE option displays the console listing continuously.

If you do not enter a file specification in the TYPE command, the system prompts for a filename with the message:

Enter filename:

Respond with the filespec of the file you want listed.

Tab characters occurring in the file named by the file specification are expanded to every eighth column position of your screen.

At any time during the display, you can interrupt the listing by pressing Ctrl-S. Press Ctrl-Q to resume the listing. Press Ctrl-C to exit back to the system.

Ensure that the file specification identifies a file containing character data.

To list the file at the printer and on the screen, type a Ctrl-P before entering the TYPE command line. To stop echoing console output at the printer, type a second Ctrl-P. The type command displays the contents of the file until the screen is filled. It then pauses until you press any key to continue the display.

Examples: A>TYPE MYPROG.PLI

This command displays the contents of the file MYPROG.PLI on your screen.

A>TYPE B:THISFILE [NOPAGE]

This command continuously displays the contents of the file THISFILE from drive B on your screen.

The USER Command

Syntax: USER {number}

Explanation: The USER command sets the current user number. When you start CP/M-86 Plus, 0 is the current user number. You can use a USER command to change the current user number to another in the range 0-15.

CP/M-86 Plus identifies every file with a user number. You can access only files identified with the current user number. However, if you mark a file in user 0 with the SYS attribute, the file can be accessed from all other user numbers.

Examples: A>USER

The system command prompts for the user number, as follows:

```
Enter user number: 5
5A>
```

The current user number is now 5 on drive A.

```
A>USER 3
3A>
```

This command changes the current user number to 3.

EOF

CPMPUG0.WS4 (= "CP/M-86 Plus User's Guide", section 0)

CP/M-86 Plus -- User's Guide

****
* First Edition: August 1983 *
****

(Edited by Emmanuel ROCHE.)

Foreword

Welcome to the world of microcomputers opened to you by your 16-bit microprocessor. Welcome also to the world of applications software accessible with your CP/M-86 Plus operating system. Digital Research designed CP/M-86 Plus especially for the 8086, 8088, or equivalent microprocessor.

What is the CP/M-86 Plus Operating System?

The CP/M-86 Plus operating system is a set of programs that manage the transfer of data between your video display screen, your floppy or hard disk system, and your printer, if you use one. The operating system allows application programs, such as text editors, financial spreadsheet programs, and the CP/M-86 Plus utility programs, to run on your computer.

CP/M-86 Plus is distributed on floppy disks. The system disk contains the operating system and the most commonly used utility programs. The other disks contain additional utilities that you can use as you become more familiar with your computer.

Purpose of this Manual

The "CP/M-86 Plus User's Guide" (hereinafter referred to as "User's Guide") describes how to use the operating system: how to enter commands at your keyboard to tell the operating system what you want it to do. For example, a command can tell CP/M-86 Plus to display the data stored on a floppy disk, or it can tell the operating system to invoke your word processing program.

Intended Audience

This "User's Guide" is intended for all users of the CP/M-86 Plus operating system. You should be familiar with the parts of your computer, how to turn it on and off, and how to handle and store floppy disks.

The version of CP/M-86 Plus on your distribution disks should be properly configured to run on your particular computer system. This configuration is done by your computer manufacturer according to the specifications described in the "CP/M-86 Plus System Guide".

Hardware Needed to Run CP/M-86 Plus

The minimum hardware requirement for CP/M-86 Plus is a computer with 64 kilobytes of random access memory (RAM) based on an 8086, 8088, or equivalent microprocessor, a video display screen with keyboard, and at least one floppy-disk drive.

Organization of This Manual

Section 1 of this manual describes how to start up your system for the first time, compose commands, make back-up copies of your disks, and get HELP at the console. Sections 2 through 4 describe additional features of CP/M-86 Plus. Section 5 explains the concept and operation of background tasking. Section 6 lists the CP/M-86 Plus utility programs in alphabetical order for easy reference. Section 7 describes ED, the CP/M-86 Plus text editor. The appendixes offer a summary of control characters, common filetypes, a glossary of commonly used computer terms, and a list of messages displayed by the operating system and its utilities.

Suggested Use of This Manual

If you are a first-time user of this operating system, read the introductory sections of this manual for an understanding of how to use CP/M-86 Plus. Then, you can refer to the command reference section (Section 6) as you use each command. Even if you have already used a previous version of CP/M, there are many new features in CP/M-86 Plus worth discovering. In either case, read the introduction to Section 6, which deals with composing command lines, before you progress with the commands themselves. All error messages displayed by CP/M-86 Plus are listed in Appendix D.

Associated Documents

There are three additional CP/M-86 Plus manuals available for purchase, which are described in the following list.

1) The "CP/M-86 Plus Programmer's Guide" presents information for application programmers who are creating or adapting programs to run under CP/M-86 Plus.

2) The "Programmer's Utilities Guide for the CP/M-86 Family of Operating

Systems" describes the CP/M assembler and related utilities that programmers use to create new CP/M-86 Plus programs.

3) The "CP/M-86 Plus System Guide" contains the information needed to adapt CP/M-86 Plus to specific hardware.

Table of Contents

Tables

EOF

CPMPUG1.WS4 (= "CP/M-86 Plus User's Guide", section 1)

(Edited by Emmanuel ROCHE.)

Section 1: Introduction to CP/M-86 Plus

This section explains how to start CP/M-86 Plus, how to enter and edit the command line, and how to make back-up copies of your CP/M-86 Plus distribution disks.

1.1 GETTING STARTED WITH CP/M-86 PLUS

The instructions presented here on how to use your system are general. Please consult the documentation provided by your system's manufacturer for information on how to turn your machine on and off, how to start the system, and how to handle disks.

First, check that the power for your disk drive and terminal is on. Next, insert the CP/M-86 Plus system disk into your initial drive, usually drive A on your system. Close the drive door. Then, press the RESET or RESTART button. This process loads CP/M-86 Plus into memory. This process is called booting, cold starting, or loading the system.

After CP/M-86 Plus is loaded into memory, a message similar to the following is displayed on your screen:

```
546K TPA -- CP/M-86 Plus for IBM AT-Compatibles
```

After this display, CP/M-86 Plus transfers control to a program called CCP (Console Command Processor). The CCP is a special system program that provides the interface between the operator and CP/M-86 Plus. This program prints the CP/M-86 Plus system prompt. It reads your command lines and processes them. The CCP displays a message similar to the following on your screen:

```
A>
```

This message is the CP/M-86 Plus system prompt. This prompt tells you that the CCP is ready to read a command line from your keyboard. In this example, the prompt also tells you that drive A is your default drive. Until you tell CP/M-86 Plus to do otherwise, it looks for program and data files on the disk in drive A.

1.2 THE COMMAND LINE

CP/M-86 Plus performs tasks according to specific commands that you type at your keyboard. A CP/M-86 Plus command line is composed of a command name, an

optional command tail, and a return keystroke. The command name identifies a command (program) to be executed. The command tail can contain extra information for the command, such as a filename or parameters. To send the command line to the CP/M-86 Plus system, you must press the ENTER key. The following example shows a valid CP/M-86 Plus command line.

```
A>dir myfile
```

The characters that you type are shown in lower case. This distinguishes them from characters that the system displays, which are in upper case. In this example, DIR is the command name and MYFILE is the command tail. The ENTER keystroke does not appear on the screen or in the example. You must remember to press the ENTER key to send a command line to CP/M-86 Plus for processing. Note that the ENTER key can be marked ENTER, RETURN, "CR", "<--+", or something similar on your keyboard. In this guide, ENTER signifies the correct key.

As you type characters at the keyboard, they appear on your screen. The single-character position indicator, called the cursor, moves to the right as you type characters. If you make a typing error, press either the BACKSPACE (or "<--") key (if your keyboard has one) or Ctrl-H to move the cursor to the left, which erases the last typed character. Ctrl is the abbreviation for the Control key. To type a control character, hold down the Ctrl key and press the required letter key. To move the cursor to the left, hold down Ctrl and press the H key. Then, you can type the correct character. CP/M-86 Plus supports other control characters that help you edit command lines. Section 3 tells how to use control characters to edit command lines and other information that you enter at your console.

You can type the command name and command tail in any combination of uppercase and lowercase letters. CP/M-86 Plus treats all letters in the command line as uppercase.

Usually, you type a command line directly after the system prompt. However, CP/M-86 Plus allows spaces between the prompt and the command name.

There are a number of commands included with CP/M-86 Plus. Section 6 lists them in alphabetical order, describes their operation and the options that are available for them, and gives examples of their use.

You can use one command to demonstrate how CP/M-86 Plus reads command lines. The DIR (directory) command tells CP/M-86 Plus to display on the screen a directory of files present on your disk. Type the DIR command name after the system prompt, without a command tail, and press ENTER.

```
A>dir
```

CP/M-86 Plus responds to this command by writing the names of all the files stored on the disk in drive A. For example, if you have your CP/M-86 Plus system disk in drive A, these filenames, among others, appear on your screen:

```
SHOW     CMD : PIP      CMD : SET      CMD
```

If you make a typing error and press ENTER before correcting your mistake,

CP/M-86 Plus prints an error message on the console. In the following example, DJR is typed instead of DIR:

```
A>djr
File not found:  DJR.CMD
```

CP/M-86 Plus tells you that it cannot find a command file called DJR.CMD.

DIR accepts a filename as a command tail. You can use DIR with a filename to see if a specific file is on the disk. For example, to check that the utility program SHOW.CMD is on your system disk, type

```
A>dir show.cmd
```

CP/M-86 Plus performs this task by displaying either the name of the file you specified, or the message:

```
File not found:  SHOW.CMD
```

Type at least one space after DIR to separate the command name from the command tail. If you do not, CP/M-86 Plus responds with the following:

```
A>dirshow.cmd
File not found:  DIRSHOW.CMD
```

1.3 BACKING UP YOUR FILES

Human or computer errors sometimes destroy valuable program or data files. By mistyping a command, for example, you can accidentally erase a program that you just created, or a data file that has been months in the making. A similar disaster can result from an electronic component failure.

Avoid losing programs and data by making copies of valuable files. Always make a working copy of any new program that you purchase, and save the original. If the program is accidentally erased from the working copy, you can easily restore it from the original. Make frequent copies of new programs or data files as you develop them.

Commands that change information recorded on your CP/M-86 Plus system disk are discussed later in this manual. First, make a few working copies of your distribution disks.

1.5 HOW TO MAKE COPIES OF YOUR CP/M-86 PLUS DISKS

This section explains how to back up your CP/M-86 Plus disks on a system that has two floppy disk drives. Please refer to your manufacturer's instructions if your system has only one floppy disk.

To back up your CP/M-86 Plus distribution disks, you need two or more floppy disks for the back-ups. The back-up disks can be new or used. You might want

to format new disks, or reformat used disks with the disk formatting program that accompanies your particular computer. If old disks are used, there must be no files on the disk that you need.

If your computer's manufacturer has provided a special program to copy disks, you might use it to make back-ups of your distribution disks. Otherwise, use the PIP utility that comes with your CP/M-86 Plus system to copy the files from your distribution disk to your work disk, after having formatted your work disk.

(The IBM PC version of CP/M-86 Plus is distributed with the FDMAINT utility, which regroup the COPYSYS, DISKCOPY, and FORMAT utilities used on previous versions. FDMAINT can format, verify, and copy whole floppy disks.)

Now, drive B contains an exact copy of the disk in drive A. Remove the original disk from drive A, and store it in a safe place. If your original remains safe and unchanged, you can easily restore your CP/M-86 Plus system if something happens to your working copy. Verify the copy in drive B by using it to restart your computer. Use this disk to copy any other distribution disks.

1.6 GETTING HELP

The HELP file command is a handy reference tool. It is part of the CP/M-86 Plus operating system. By typing in the following command, you can see a list of all the topics for which information exists in the system:

```
A>help
```

After displaying a list of all the topics, the HELP program then prompts you as follows for a topic:

```
HELP>
```

You then enter the topic you want information on, and press ENTER. You are given a brief explanation, followed by the names of the subtopics available. By entering the name of the subtopic preceded by a period ("."), you get additional specific information. See the HELP command description in Section 6 for a full description of this command.

EOF

CPMPUG2.WS4 (= "CP/M-86 Plus User's Guide", section 2)

(Edited by Emmanuel ROCHE.)

Section 2: Files, Disks, and Drives

The most important task of CP/M-86 Plus is to access and maintain files on your disks. With CP/M-86 Plus, you can create, read, write, copy, and erase disk files. This section tells you what a file is, how to create, name, and access a file, and how files are stored on your disks. It also tells you how to change disks, and change the default drive.

2.1 WHAT IS A FILE?

A file is usually a collection of related information stored on a disk. Every file must have a unique name, because CP/M-86 Plus uses that name to access that file. A directory is also stored on each disk. The directory contains a list of the filenames stored on that disk, and the locations of each file on the disk.

Usually, there are two kinds of files: command (program) files and data files. A command file contains an executable program (a series of instructions that the computer follows step-by-step). A data file is usually a collection of information: a list of names and addresses, the inventory of a store, the accounting records of a business, the text of a document, or similar information. Your computer cannot execute names and addresses, but it can execute a program that prints names and addresses on mailing labels.

2.2 CREATING FILES

There are many ways to create a file. One way is to use a text editor. The CP/M-86 Plus text editor, ED, (described in Sections 6 and 7) can create a file and assign it the name that you specify. You can also create a file by copying an existing file to a new location, perhaps renaming it in the process. Under CP/M-86 Plus, you can use the PIP command to copy and rename files. Finally, some application programs create and delete output files as they process input files.

2.3 NAMING FILES

Throughout this manual, a specific file is identified by using a string of characters called a "file specification". A file specification can be simply a one- to eight-character filename, such as:

MYFILE

Or, a file specification can have four parts: a drive specifier, a filename, a filetype, and a password.

The drive specifier is a single letter (A-P) followed by a colon (":"). Each drive in your system is assigned a letter. When you include a drive specifier as part of the file specification, you are telling CP/M-86 Plus that the file is stored on the disk currently in that drive. For example, if you enter the following command, you specify file MYFILE on drive B:

B:MYFILE

The filename can be from one to eight characters. When you make up a filename, give it a name that tells you what the file contains. For example, if you have a list of customer names for your business, you can name the file:

CUSTOMER

As you use your computer with CP/M-86 Plus, you will see that files fall naturally into categories. To help you identify files belonging to the same category, CP/M-86 Plus allows you to add an optional one- to three-character extension, called a "filetype", to the filename. When you add a filetype to the filename, separate the filetype from the filename with a period ("."). Try to use three letters that tell something about the file's category. For example, you might add the following filetype to the file that contains a list of customer names:

CUSTOMER.NAM

When CP/M-86 Plus displays file specifications in response to a DIR command, it adds blanks to short filenames, so that you can compare filetypes in the same column. The command files that CP/M-86 Plus loads into memory and executes have different filenames, but all have the filetype CMD.

In CP/M-86 Plus, you can protect your files by assigning them a password. To access the file, you must specify the password that you assigned to the file. When you use the file, you can specify the password by adding a password as an optional part of the file specification. The password can be from one to eight characters. If you include a password, separate it from the filetype (or filename, if no filetype is included) with a semicolon (";"), as shown in the following example.

CUSTOMER.NAM;ACCOUNT

A method for setting a default password for the system to use if you do not supply a password as part of the file specification, is discussed in Section 6.

Create filenames, filetypes, and passwords from letters and numbers. You must not use the following characters in filenames, filetypes, or passwords because they have special meanings for CP/M-86 Plus:

< > = , ! * ? / $ [ ] ( ) . : ; \

A complete file specification containing all possible elements consists of a drive specifier, a filename, a filetype, and a password, all separated by their appropriate delimiters, as in the following example:

```
A:DOCUMENT.LAW;SUSAN
```

2.4 DO YOU HAVE THE CORRECT DRIVE?

The part of your computer that holds the floppy or hard disk is called the drive. In CP/M-86 Plus, drives are named A, B, ... P, but your system probably uses only a few of these drives. When you first turn on your computer's power, the system displays the default drive identified in the system prompt:

```
A>
```

When you type a file specification without a drive specifier in the command tail, CP/M-86 Plus commands look for the file in the drive named by the system prompt (also referred to as the "default drive"). For example, if you type the command:

```
A>DIR SHOW.CMD
```

DIR looks in the directory of the disk in drive A for SHOW.CMD. If you have another drive, B for example, you need a way to tell CP/M-86 Plus to access the disk in drive B. For this reason, CP/M-86 Plus lets you precede a filename with a drive specifier. For example, in response to the command:

```
A>DIR B:MYFILE.LIB
```

CP/M-86 Plus looks for the file MYFILE.LIB in the directory of the disk in drive B. When you give a command to CP/M-86 Plus, note which disk is in the default drive. Some application programs require that the data files they access be stored in the default drive.

You can also precede a command name with a drive specifier. For example, if you enter the following command:

```
A>B:SHOW
```

CP/M-86 Plus looks in the directory of the disk in drive B for the file SHOW.CMD. If CP/M-86 Plus finds SHOW.CMD on drive B, it loads it into memory, and executes it.

If you must access many files on the same drive, you might want to change the default drive, so that you do not need to repeatedly enter a drive specifier. To change the default drive, enter the drive specifier next to the system prompt, and press ENTER. In response, CP/M-86 Plus changes the system prompt to display the new default drive:

```
A>B:
B>
```

Unlike the filename and filetype, which are stored in the disk directory, the drive specifier for a file changes as you move the disk from one drive to another. Therefore, a file has a different file specification when you move a disk from one drive to another.

2.5 DO YOU HAVE THE CORRECT USER NUMBER?

CP/M-86 Plus assigns to all files a user number from 0 to 15. You can use this facility to divide your files into 16 groups. When you create a file, the system assigns the current user number to this file. The system displays the current user number with the drive letter in the system prompt. For example, the system prompt can look like this:

```
3B>
```

This indicates that 3 is the current user number. When the system does not display a user number in the prompt, the current user number is 0. This is usually the case when the system is loaded and displays the following prompt:

```
A>
```

All of the files that you create are filed by the system under the current user number. If you always run under user 0, then all of your files will exist under user 0.

You can change the current user number by using the USER command. For example, if you want to create or access files under user 5, enter the following:

```
A>user 5
```

and the system responds with

```
5A>
```

to indicate that you are now running under user 5. You can change both the user number and the drive by entering the new user number and the drive specifier together at the system prompt:

```
A>3b:
3B>
```

Although the system allows you to create or modify files only under the current user number, it allows you to read files under user 0 when they have the system attribute. See "System and Read-Only Attributes" in this section for a description of the SYS attribute. This facility eliminates the need to copy all of the files that you need to every user number. For example, if you are running under user 5, and you enter the following command:

```
5A>type abc.lst
```

the system first looks for the file ABC.LST under user 5. If it does not find

it there, the system looks for the file ABC.LST with the SYS attribute under user 0. If the system finds the file, TYPE displays the contents of the file on the screen.

Some CP/M-86 Plus commands restrict their file access to the current user number. For example, a DIR command with no special options displays only the files that were created under the current user number. You can use the DIR options to show files under other users.

Note that you can move files from one user number to another with the PIP command. See Section 6 for details on the PIP command.

2.6 ACCESSING MORE THAN ONE FILE

Certain CP/M-86 Plus commands can select and process several files when special wildcard characters are included in the filename or filetype. A file specification containing wildcards is called a "wildcard filespec" and can refer to more than one file, because it gives CP/M-86 Plus a pattern to match. CP/M-86 Plus searches the disk directory, and selects any file whose filename or filetype matches the pattern.

The two wildcard characters are ?, which matches any single letter in the same position, and *, which matches any character at that position and any other characters remaining in the filename or filetype. The following rules apply:

- A ? matches any character in a name, including a space character.

- An * must be the last, or only, character in the filename or filetype. CP/M-86 Plus replaces an * with as many ? characters as are needed to fill to the end of the filename or filetype.

- When the filename to match is shorter than eight characters, CP/M-86 Plus treats the name as if it ended with spaces.

- When the filetype to match is shorter than three characters, CP/M-86 Plus treats the filetype as if it ended with spaces.

Suppose, for example, you have a disk that contains the following six files:

A.CMD, AA.CMD, AAA.CMD, B.CMD, A.DAT and B.DAT

The following wildcard specifications match all, or a portion of, these files:

.	is treated as ????????.??? and matches all six names
*.CMD	matches the first four names
?.CMD	matches A.CMD and B.CMD
?.*	matches A.CMD, B.CMD, A.DAT, and B.DAT
A?.CMD	matches A.CMD and AA.CMD
A*.CMD	matches A.CMD, AA.CMD, and AAA.CMD

Remember that CP/M-86 Plus uses wildcard patterns only while searching a disk directory, and therefore wildcards are valid only in filenames and filetypes.

You cannot use a wildcard character in a drive specifier. You also cannot use a wildcard character as part of a filename or filetype when you create a file, because each filename must be unique.

2.7 HOW TO PROTECT AND MONITOR YOUR FILES

Under CP/M-86 Plus, you can protect files from accidental change and from unauthorized access. You can specify how your files are displayed in response to a DIR command, and monitor when your files were last accessed or modified. CP/M-86 Plus supports these features by assigning the following to files:

- System and Read-Only attributes
- time/date stamps
- passwords

2.7.1 System and Read-Only Attributes

File attributes control how programs access files. When you create a file, CP/M-86 Plus gives it two attributes, DIR for directory and RW for Read/Write. You can change these attributes with the SET command.

The first attribute can be set to either DIR (Directory) or SYS (System). This attribute controls whether CP/M-86 Plus displays the filename in response to a DIR command or DIRSYS command. You can display the name of a file set to the DIR attribute with a DIR command. You can display the name of a file set to the SYS attribute with the DIRSYS command. DIR and DIRSYS commands display only the filenames created under the current user number, unless you specify the options available with the DIR command. See the description of DIR in Section 6 for information on these options.

If you store a file under user 0 and set the SYS attribute, you can access the file from any other user number. For example, suppose that you organize your files so your memos are in user 8 and your correspondence in user 9. If you want to use a text editor to edit those files, you can place a single copy of the text editor program in user 0 with the SYS attribute set. Otherwise, you need a copy of the text editor both in user 8 and user 9. This feature eliminates the need to maintain separate copies of commonly used files in each user number.

The second file attribute can be set to either RW (Read/Write) or RO (Read-Only). If a file is set to Read-Only, any attempt to write data to that file produces an error message. Therefore, you can use the RO attribute to protect important files. A file with the RW attribute can be read, written to, or erased at any time, unless the entire disk is set to Read-Only or the disk is physically write-protected.

See the description of the SET utility in Section 6 for more information on setting attributes.

2.7.2 Date/Time Stamping

If you use date/time stamps, you can locate the most recent copy of a file, by checking when it was last updated or changed. You can choose to have the system tell you either when you created the file, or when you last accessed the file. Use the SET command to enable date/time stamping, and the DIR command with the DATE option to display date/time stamp of a file.

You can use the following SET commands to enable date/time stamping on a disk, but you must choose between ACCESS and CREATE. If you choose ACCESS, the stamp records the last time the file was accessed. If you choose CREATE, the stamp records when the file was created. If you select UPDATE, the stamp also records the last time the file was updated.

```
A>SET [ACCESS=ON]
A>SET [CREATE=ON]
A>SET [UPDATE=ON]
```

Files created on or copied to a disk with date/time stamping are automatically stamped. The DATE command allows you to display and reset the date and time that CP/M-86 Plus is using. For a complete discussion of date/time stamping, see the descriptions of the SET and INITDIR commands in Section 6.

2.7.3 Passwords

Passwords allow you to protect your files from access by other operators.

The SET utility allows you to enable password protection on a drive, assign a password to the disk (so that unauthorized users cannot disable password protection on that disk), and assign passwords to specific files that are already created. You can assign passwords to all program and data files. A command line can require the entry of two passwords to execute: one password to access the command program, and a second password to access the file specified in the command tail. The following is an example command line with two passwords:

```
A>BACKUP;PASSWD1 BAKINPUT.DAT;PASSWD2
```

Because CP/M-86 Plus is new and password protection is not supported in previous versions, some CP/M-86 Plus programs do not recognize passwords in the command tail. Rather than specifying a password in the command tail, you can set a default password. The default password remains in effect until you change it again using the SET utility. The default password and the specified password (if any) are both verified against the actual password of the file. If either one matches the file's password, then CP/M-86 Plus allows access to the password-protected file. Use the SET utility to establish a default password before you access a password-protected file. For example,

```
A>SET [DEFAULT=MYPASS]
A>PROGA
```

is functionally equivalent to

```
A>PROGA;MYPASS
```

Note: You can use the DIR command with the [FULL] option to display the files and their attributes, date/time stamps, and password-protection modes.

2.8 HOW ARE FILES STORED ON A DISK?

CP/M-86 Plus records the filename, filetype, password, user number, and attributes of each file in a special area of the disk called the directory. In the directory, CP/M-86 Plus also records which areas of the disk belong to which file.

CP/M-86 Plus allocates directory and storage space for a file as information is added to the file. When you erase a file, CP/M-86 Plus reclaims storage in two ways: it makes the file's directory space available to catalog a different file, and frees the file's storage space for later use. You do not have to tell CP/M-86 Plus how big your file might be, because it automatically allocates more storage for a file as needed, and releases the storage for reallocation when the file is erased. Use the SHOW command to determine how much space remains on the disk.

2.9 CHANGING FLOPPY DISKS

CP/M-86 Plus cannot do anything to a file unless the disk that holds the file is inserted into a drive, and the drive door is closed. When a disk is in a drive, it is online and CP/M-86 Plus can access its directory and files.

You might need to take a disk out of a drive sometime, and to insert another that contains different files. You can replace a disk when you see the system prompt at your console, unless a program in the background is reading or writing to the disk in the drive.

You can also remove a disk and insert a new one when an application program prompts you. This situation can occur, for example, when the data that the program uses does not fit on one floppy disk.

Note: Never remove a disk if any program is reading from or writing to it.

2.10 PROTECTING A DISK OR DRIVE

Under CP/M-86 Plus, you can set a drive to Read-Only, just as you can set a file to Read-Only. The default state of a drive is Read/Write. You can use the SET command described in Section 6 to set a drive to Read-Only. To return the drive to Read/Write, use the SET command or press Ctrl-C following the system prompt. Also, if you change disks in the drive after setting it to Read-Only, then CP/M-86 Plus changes the drive back to Read/Write.

EOF

CPMPUG3.WS4 (= "CP/M-86 Plus User's Guide", section 3)

(Edited by Emmanuel ROCHE.)

Section 3: Console and Printer

This section describes how CP/M-86 Plus communicates with your console and printer. You can start/stop console and printer output, edit commands that you enter at your console, and redirect console and printer input/output. This section also explains the concept of logical devices under CP/M-86 Plus.

3.1 CONTROLLING CONSOLE OUTPUT

Most CP/M-86 Plus utilities support automatic paging at the console. If the output fills the screen, the display automatically pauses. When this pause occurs, the utility prompts you to press ENTER to continue. You can use the SETDEF command to turn off the automatic paging feature. You can control the number of lines automatically displayed by some of the CP/M commands on your terminal with the DEVICE command. These settings remain in effect until you change them, or until you turn off the power.

Some application programs do not support paging. If an especially long display scrolls off the top of your screen before you have a chance to study it, press Ctrl-S to cause the display to pause. When you are ready to resume the display, press Ctrl-Q.

3.2 CONTROLLING PRINTER OUTPUT

You can also use a control command to echo console output to the printer. To start printer echo, press Ctrl-P. To stop printer echo, press Ctrl-P again. While printer echo is in effect, any characters that appear on your screen are listed at your printer.

You can use printer echo with a DIR command to make a list of files stored on a floppy disk. You can also use Ctrl-P to make a hard copy of part of a file. Use a TYPE command to start a display of the file at the console. When the display reaches the part that you want to print, press Ctrl-P to enable printer echo. You can use another Ctrl-P sequence to terminate printer echo.

3.3 CONSOLE LINE EDITING

CP/M-86 Plus allows you to edit your command line. Using the line-editing control characters listed in Table 3-1, you can move the cursor left and right to insert and delete characters in the middle of a command line. You do not

have to retype everything to the right of your correction. You can press ENTER when the cursor is in any position in the command line; CP/M-86 Plus reads the entire line. You can also recall a command that is already processed by the system, and reedit and reprocess the command.

In the following sample session, assume that you mistype ERA and that CP/M-86 Plus returns an error message. You can recall the erroneous command line by pressing Ctrl-W. To correct the error, use the following control characters. The underline indicates the position of the cursor.

```
A>EFA A:MYFILE.TST     (ERA mistyped)
File not found         (Error Message)
A>EFA MYFILE.TST_      (Ctrl-W to recall the line)
A>EFA MYFILE.TST       (Ctrl-B to move cursor to beginning of line)
A>EFA MYFILE.TST       (Ctrl-F to move cursor right, under "F")
A>EA MYFILE.TST        (Ctrl-G to delete error)
A>ERA MYFILE.TST       (Type letter R to correct the command name,
                        and press ENTER)
```

To process the corrected command line, press ENTER, even though the cursor is in the middle of the line. A ENTER keystroke, or one of its equivalent control characters, not only processes the command, but also stores the command in a buffer so you can recall it for editing or reprocessing by pressing Ctrl-W.

When you insert a character in the middle of a line, characters to the right of the cursor move to the right. If the line becomes longer than your screen is wide, characters disappear off the right side of the screen. These characters are not lost. They reappear if you delete characters from the line, or if you press Ctrl-E when the cursor is in the middle of the line. Ctrl-E moves all characters to the right of the cursor to the next line on the screen.

The following table gives a complete list of line-editing control characters.

Table 3-1. CP/M-86 Plus Line-editing Control Characters

Format: Character
Meaning

RUB/DEL
Removes and echoes the last character if at the end of the line; otherwise, deletes the character to the left of the current cursor position.

Ctrl-A
Moves the cursor one character to the left.

Ctrl-B
Moves the cursor to the beginning of the command line, without having any effect on the contents of the line. If the cursor is at the beginning, Ctrl-B moves it to the end of the line.

Ctrl-E
Forces a physical ENTER, but does not send the command line to CP/M-86 Plus. Moves the cursor to the beginning of the next line without erasing the

previous input.

Ctrl-F
Moves the cursor one character to the right.

Ctrl-G
Deletes the character indicated by the cursor. The cursor does not move.

Ctrl-H
Deletes a character to the left of the cursor, and moves the cursor left one character position.

Ctrl-I
Moves the cursor to the next tab stop. Tab stops are automatically set at each eighth column. Has the same effect as pressing the TAB key ("-->|").

Ctrl-J
Sends the command line to CP/M-86 Plus, and returns the cursor to the beginning of the line. Has the same effect as ENTER or Ctrl-M.

Ctrl-K
Deletes all characters from the cursor to the end of the line.

Ctrl-M
Sends the command line to CP/M-86 Plus, and returns the cursor to the beginning of the line. Has the same effect as ENTER or Ctrl-J.

Ctrl-R
Retypes the command line to the left of the cursor. Places a # at the current cursor location, moves the cursor to the next line, and retypes any partial command you have typed so far.

Ctrl-U
Discards all the characters in the command line, places a # at the current cursor position, and moves the cursor to the next line. However, you can use a Ctrl-W to recall any characters that were to the left of the cursor when you pressed Ctrl-U.

Ctrl-W
Recalls and displays previously entered command line both at the operating system level and within processing programs, if the Ctrl-W is the first character entered after the prompt. If the command line contains characters, Ctrl-W moves the cursor to the end of the command line. If you press ENTER, CP/M-86 Plus executes the recalled command.

Ctrl-X
Discards all the characters to the left of the cursor, and moves the cursor to the beginning of the current line. Ctrl-X saves any characters to the right of the cursor.

Some control characters have the same meaning. For example, Ctrl-J and Ctrl-M have the same effect as pressing ENTER; all three send the command line to CP/M-86 Plus for processing. Also, Ctrl-H has the same effect as pressing the BACKSPACE key ("<--"). Notice that, when a control character is displayed on

your screen, it is preceded by a circumflex, ^. For example, Ctrl-D appears as ^D on your screen.

3.4 REDIRECTING CONSOLE INPUT AND OUTPUT

Use console redirection when you want to copy to a file the information that a program displays on your screen, or when you have created a file that you want to use as console input to a program.

3.4.1 Copying Console Output to a File

When redirecting console output to a disk file, all information normally displayed on your screen by the program is written to a disk file. You can create a file containing the information displayed on a screen if you want to:

- use this information as input to another program.

- edit this information, or add headings and other data to create a report.

- make a number of printed copies of this information.

There are two ways to direct program output to be copied to a file. The simplest method is to include in the command line the redirection symbol > and the name of the output file. CP/M-86 Plus recognizes the > as a symbol that means copy the output of the program to the specified file. You can also invoke the PUT command described in Section 6 to direct console or printer output to a disk file.

When you copy the output of the DIR command to a file, you can create a file containing a directory of all files on a disk, as follows:

```
A>DIR

A: FILENAME TEX : FRONT    TEX : FRONT    BAK : ONE      BAK : THREE    TEX
A: FOUR     TEX : ONE      TEX : LINEDIT  TEX : EXAMP1   TXT : TWO      BAK
A: TWO      TEX : THREE    BAK : EXAMP2   TXT

A>DIR > DEMO.DIR
```

(Note that the console output do NOT appear on the screen. If you need to see the console output (for example, while using interactively a program), use the PUT command.)

```
A>TYPE DEMO.DIR

A: FILENAME TEX : FRONT    TEX : FRONT    BAK : ONE      BAK : THREE    TEX
A: FOUR     TEX : ONE      TEX : LINEDIT  TEX : EXAMP1   TXT : TWO      BAK
A: TWO      TEX : THREE    BAK : EXAMP2   TXT
```

3.4.2 Reading Console Input From a File

When you redirect console input from a file, the operator input usually entered at the keyboard is read from a file. The program continues to read input from the file until the program does not require further input from the console, or until the file is exhausted. If you do not include all program input in the file, the program attempts to read the remaining input from the console. Note that all console input must be included in the file if you are going to run the program in the background and use the console to enter input for another program. See Section 5 for a discussion of running programs in the background. You can redirect console input to come from a file when you:

- use a file created by another program as console input to this program.

- use an editor to create input to this program, because you run the program often and type the same initial input at the keyboard.

- want to run this program in the background, to free the console to run another program.

There are two ways to indicate that you want the program to read a disk file for console input, instead of reading the keyboard. The simplest method is to include in the command line the redirection symbol < and the name of the input file. You can also use the GET command, described in Section 6, to redirect console input.

In the following example, the file PIP.DAT is typed at the console to show you what it contains. Then PIP is invoked with input from the file PIP.DAT:

```
3A>type pip.dat
b:=front.tex
b:=one.tex
b:=two.tex

3A>pip < pip.dat
CP/M-86 Plus PIP Version 3.1
*b:=front.tex
*b:=one.tex
*b:=two.tex
*ENTER        (=Press the ENTER key on your keyboard.)
```

PIP reads console input from the file PIP.DAT until it has read and processed all commands in PIP.DAT. Then PIP reads further input from the console. In this example, ENTER is entered at the console.

You can also use the GET and PUT commands to redirect console input and output. They have additional options that you can use to indicate that console input or output is not to be echoed to the console, and that system commands are included in the input or output files. See the descriptions of GET and PUT in Section 6 for a complete description of their options, and for more ways to use redirected input and output.

3.5 ASSIGNING LOGICAL DEVICES

Most CP/M-86 Plus computer systems have a console with a keyboard and screen display, and perhaps a printer. To keep track of these different input and output devices, CP/M-86 Plus assigns different physical devices to logical devices in the system.

A logical device represents a class of physical devices. For example, both the keyboard and auxiliary port are physical devices that can provide input. Console input, therefore, is a logical device that can be assigned to a physical device like the keyboard. Similarly, console output can be assigned to a physical device such as the screen. When a logical device is not assigned to anything specific, it is said to be assigned to a null, or dummy, device.

The following table gives the names and types of CP/M-86 Plus logical devices. It also shows the physical devices assigned to these logical devices in the standard CP/M-86 Plus system.

Table 3-2. CP/M-86 Plus Logical Device Assignments

Logical Device Name	Logical Device Type	Physical Device Assignment
CONIN:	Console input	Keyboard
CONOUT:	Console output	Screen
AUXIN:	Auxiliary input	Null
AUXOUT:	Auxiliary output	Null
LST:	List output	Printer

Typically, you can assign a number of physical devices to any given logical device. The following table shows a number of possibilities.

Table 3-3. Typical Device Assignments

Logical Device Name	Logical Device Type	Typical Physical Devices
CONIN:	Console Input	Keyboard
CONOUT:	Console Output	Screen, Printer
AUXIN:	Auxiliary Input	Modem, Lightpen, Joystick, Mouse, and so on.
AUXOUT:	Auxiliary Output	Modem, Printer, Plotter, and so on.
LST:	List Output	Printer, Screen

If you use your computer for a range of tasks, you might want to add different kinds of devices to your system, for example, a line printer, a modem, a light pen, or even a joystick for playing games.

In some implementations of CP/M-86 Plus, you can change the standard

assignments with a DEVICE command. If your system supports the DEVICE command, you can, for example, assign AUXIN: and AUXOUT: to a modem, so that your computer can communicate with other computers over the telephone. However, for a physical device to be supported in the system, a driver (code that specifically addresses that physical device) must be written. The manufacturer of a given device usually provides a driver for the device. Otherwise, the end user must be able to program the necessary driver, to enable that device to be added to the system. Check with your system manufacturer or peripheral device manufacturer to ensure that the configuration you are attempting actually works.

EOF

CPMPUG4.WS4 (= "CP/M-86 Plus User's Guide", section 4)

(Edited by Emmanuel ROCHE.)

Section 4: CP/M-86 Plus Command Line

This section describes

- the CP/M-86 Plus command line
- the procedure for finding program files and data files
- how to enter multiple commands in a command line
- how to stop a running program

4.1 THE CP/M-86 PLUS COMMAND LINE

The command line allows you to specify the program that you want to run, as well as some additional information that the program might need. The Console Command Processor (CCP.CMD) processes the command line. You enter the command line after the system prompt, and press ENTER. The command line has two parts:

- the command keyword, or the name of the program you want to invoke
- the command tail, or information for that program

For example, if you enter:

```
A>DIR B:*.CMD[FULL]
```

DIR is the command keyword, and B:*.CMD[FULL] is the command tail. In most cases, the command keyword that you enter is the name of a file with a filetype of CMD. However, the filetype of CMD does not have to be typed. There are special cases where the command that you specify is internal to the CCP: DIR, ERA, REN, TYPE, USER, and FORE. However, some of these commands have options that require a related CMD file. Therefore, CMD files like DIR.CMD are supplied with CP/M-86 Plus.

If you get an error message in the form:

```
File not found: FILESPEC.CMD
```

the CCP cannot find the CMD file required to accomplish the task. The CMD file it cannot find might be an independent utility, or it might be a CMD file related to an internal CCP command.

The command tail includes information that the program uses, and can consist of a file specification used by a CP/M-86 Plus command and/or an option list. The file specification used by CP/M-86 Plus commands (described in detail in Section 6) consists of the following four parts:

1) drive specifier -- the name of a drive
2) filename -- the name of a file, or group of files
3) filetype -- the family name of the file, or group of files
4) password -- the secret word that allows access to your file

For example, if you enter the following command:

```
B>TYPE A:PAYROLL.DAT;ROSEBUD[NOPAGE]
```

then:

- B> is the system prompt
- TYPE is the command keyword
- A:PAYROLL.DAT;ROSEBUD[NOPAGE] is the command tail
- A:PAYROLL.DAT;ROSEBUD is the file specification where:
  - A is the drive specifier
  - PAYROLL is the filename
  - DAT is the filetype
  - ROSEBUD is the password
- [NOPAGE] is the option list

In the filespec, a colon (":") follows the drive specifier, a period (".") comes between the filename and the filetype, and a semicolon (";") separates the password from the rest of the specification. Left and right square brackets ("[" and "]") enclose the options for this command.

Some programs allow you to select special tasks that you want to perform by including an options list in the command line. Each program looks for options in a different manner, so refer to Section 6 if you use a program supplied with CP/M-86 Plus.

4.2 HOW CP/M-86 PLUS SEARCHES FOR PROGRAM AND DATA FILES

Sometimes, you might enter a command line and get an error message because either CP/M-86 Plus (the CCP) cannot find the command that you specified, or the program cannot find a file specified in the command tail. When this occurs, CP/M-86 Plus is probably looking in the wrong place for one of the files. This section explains how CP/M-86 Plus searches for program files and data files. The search for a file used as a command can be very different from a search for a file used as data for a command.

A command keyword specifies a program file. A program file contains the instructions that allow a computer to perform a certain task. This task can be word processing, general ledger, data file manipulation, or something else. To identify a program file, there are two questions to ask:

1) Is the filename used as a command keyword?
2) Does the file have the filetype CMD?

If the answer to both questions is yes, then CP/M-86 Plus is looking for a program file.

The command tail can include a data file. A data file contains information used by a program. This information can be a list of customer names, an inventory list, a list of messages, or a program file. Data files can have any filetype, including CMD. Therefore, a program file can be used as a data file by another program. For example,

```
A>DUMP B:PIP.CMD
```

Here, the DUMP.CMD file is specified as a program, and the PIP.CMD file is specified as a data file. CP/M-86 Plus looks for these files in different ways, even though both have the same filetype, and both can be used as commands. Because PIP.CMD is specified in the command tail, it is treated as a data file. In the following example, PIP.CMD and DUMP.CMD reverse roles.

```
A>PIP B:=DUMP.CMD
```

Here, the PIP.CMD file is specified as a program, and the DUMP.CMD file is specified as a data file. The B:=DUMP.CMD is information for the PIP program. The way CP/M-86 Plus looks for a file is determined by the way you want to use the file.

4.2.1 Finding Data Files

When you enter a command line, the CCP passes the command tail to the program identified by the command. If the command tail contains a file specification, the program calls CP/M-86 Plus to search for the data file. If CP/M-86 Plus cannot find the data file, the program displays an error message at the console.

If you do not include a drive specifier with the filename in a command tail, CP/M-86 Plus searches the directory of the current user number on the default drive. If the file is not there, CP/M-86 Plus also looks for the file with the SYS attribute in the directory of user 0 on the default drive. If CP/M-86 Plus finds the file under user 0, it allows the program (Read-Only) access to the file. For example, if you enter the following command line:

```
3A>TYPE MYFILE.TXT
```

CP/M-86 Plus first searches the directory for user 3 on drive A. If it does not find MYFILE.TXT there, it searches the directory of user 0 on drive A for MYFILE.TXT marked with the SYS attribute. If the file is not in either directory, CP/M-86 Plus returns control to TYPE, which then displays a message saying that the file was not found.

The search procedure is basically the same if you include a drive specifier with the filename. CP/M-86 Plus first looks in the directory of the current user number on the specified drive. Then, if it does not find the file, it looks in the directory for user 0 on the specified drive for the file with the SYS attribute. If CP/M-86 Plus does not find the data file after these two searches, the program displays an error message. Note that programs like DIR restrict their search to the current user number.

4.2.2 Finding Program Files

The search procedure for a program file can be very different from a search for a data file. You can use the SETDEF command described in Section 6 to define the search procedure that you want CP/M-86 Plus to follow when it is looking for a program file. With SETDEF, you can ask CP/M-86 Plus to make as many as 16 searches when you do not include a drive specifier before the command keyword. CP/M-86 Plus searches for program files when you have included a drive specifier with the command keyword.

If you include a drive specifier before the command keyword, you are telling CP/M-86 Plus precisely where to look for the program file. Therefore, CP/M-86 Plus searches only two locations: the directory under the current user on the specified drive, and then under user 0 on the specified drive for the same file marked with the SYS attribute. For example, if you enter

```
4C>A:SHOW [SPACE]
```

CP/M-86 Plus looks for the file SHOW.CMD on drive A, user 4, and then on drive A, user 0 for the file with the SYS attribute. If CP/M-86 Plus does not find SHOW.CMD, it displays an error message.

If you do not include a drive specifier before the command, the sequence CP/M-86 Plus uses to search directories is called a drive chain. If you did not use the SETDEF command to define a search procedure, there is only one drive in your chain, the default drive. Then, CP/M-86 Plus looks for the program file specified by your command on the default drive. It looks under the current user number, and then under user 0 for the same file marked with the SYS attribute. For example, if you enter

```
7E>SHOW [SPACE]
```

CP/M-86 Plus searches the following locations for the file SHOW.CMD:

1) drive E, user 7
2) drive E, user 0

Remember that a SHOW.CMD file under user 0 must be marked with the SYS attribute, or else CP/M-86 Plus cannot find it.

You can use a SETDEF command to define the drives that you want to have CP/M-86 Plus search for command files. When you define your own drive chain, include the default drive and the drive that contains your most frequently used utilities. For example, assume that you defined your drive chain as @ (the default drive) and drive A. When you enter the command:

```
2D>SHOW [SPACE]
```

CP/M-86 Plus looks for SHOW.CMD in the following sequence:

1) drive D, user 2
2) drive D, user 0

3) drive A, user 2
4) drive A, user 0

Any drive chain that you specify with SETDEF remains in effect until you restart or reset the system.

4.3 EXECUTING MULTIPLE COMMANDS

To enter multiple commands at the system prompt, separate each command and associated command tail from the next command with an exclamation point ("!"). When you complete the sequence, press ENTER. CP/M-86 Plus executes your commands in order:

```
3A>dirsys! dir examp*.*! show [space]
NON-SYSTEM FILE(S) EXIST
3A>dir examp*.*
A: EXAMP7      : EXAMP1  TXT : EXAMP3      : EXAMP2  TXT : EXAMP4
A: EXAMP5      : EXAMP6
3A>show space

A: RW, Space:    3,344k
```

4.4 STOPPING A RUNNING PROGRAM

You can use Ctrl-C to stop a running program. Most application programs that run under CP/M-86 Plus can be stopped by a Ctrl-C. All utilities supplied with CP/M-86 Plus can be stopped by a Ctrl-C, except the BACK command.

You can also use Ctrl-C to reset the disk system unless you are running a program in the background, as described in Section 5. When you press Ctrl-C and the cursor is at the system prompt, CP/M-86 Plus logs out all the active drives, then logs in the default drive. The active drives are all drives accessed since the last cold or warm start. A SHOW [SPACE] command displays the remaining space on all active drives. In the following example, SHOW [SPACE] indicates that three drives are active. However, if you press Ctrl-C immediately after this display and then enter another SHOW [SPACE] command, only the space for the default drive, A, is displayed.

```
A>SHOW [SPACE]
A: RW,  Space:    9,488k
B: RO,  Space:    2,454k
C: RO,  Space:    1,665k
A>^C
```

```
A>SHOW [SPACE]
A: RW,  Space:    9,488k
```

When you are running a program in the background, you cannot stop this program by entering a Ctrl-C. To stop the program, you must use the TASK command described in Section 6.

EOF

CPMPUG5.WS4 (= "CP/M-86 Plus User's Guide", section 5)

(Edited by Emmanuel ROCHE.)

Section 5: Background Programs

CP/M-86 Plus allows you to run up to four programs at the same time. However, only one program can use the console and the keyboard. This program is called the "foreground program". The other programs, those not using the console and keyboard, are called "background programs". Use the BACK command to run programs in the background.

For example, WS4 has a command to print a file, or chapter, or whole book, from the command line, without appearing on screen (WS filename PX). Normally, you cannot do anything else while WS4 is printing. With BACK, you can continue to use your computer, as long as you don't use the device used by the background program (in this example, the printer).

There are three background programs possible, and the standard BIOS of CP/M-86 Plus for the IBM AT-Compatibles recognizes three I/O devices (besides the CONSOLE:, which regroups the screen and the keyboard): AUXIN:, AUXOUT:, and LST:. LST: is an Output-only, Parallel port (so, is specialized for printing purposes), while the two AUXILIARY: ports are Input/Output and Serial. So, the possibilities are only limited by your imagination...

Just develop your applications using the screen and keyboard as usual and, when it is debugged, all its input will come from the INFILE (replacing the keyboard), and all its output will go to the OFILE (replacing the screen) (See "The BACK Command" in Section 6). As usual, everything is simpler if you only use ASCII files.

5.1 REQUIREMENTS FOR EXECUTING PROGRAMS IN THE BACKGROUND

If you want to run a program in the background, you must follow these procedures:

- If the background program requires entries from the keyboard, you must create a file that contains all those entries. Refer to "Reading Console Input From a File" in Section 3. If you do not redirect console input from a file, or if the file does not contain all input, the background program is terminated by the system.

- You must run only one program that uses the printer. If you try to run two programs that use the printer at the same time, the printout from the two programs is intermixed.

- You must ensure that your background program does not change a file that another program is reading or changing.

- You must ensure that your program does not read a file that another program is changing. Note that two or more programs can read the same file at the same time.

5.2 USING THE BACK COMMAND

To run a program in the background, you must use the BACK command. The BACK command allows you to specify the following:

- The name of the command that you want to run in the background.

- The command tail for the command that you want to use.

- The name of the file containing the console input that the program needs.

- The name of the file to receive all the information that the program usually sends to the console, including any error messages. This file is called the BAC file, since this is the filetype that BACK uses when it creates files.

5.3 PRECAUTIONS FOR RUNNING BACKGROUND PROGRAMS

Use the following precautions to avoid complications while using background programs:

- Make GET.RSX and PUT.RSX available through your drive search chain, since those two RSXs are used by BACK to communicate with the background programs.

- Do not change removable disks, unless you are certain that no background program is using the disk.

- Periodically TYPE the BAC file, to check the progress of your background programs. (BAC files are standard ASCII files.)

- To use two printers, assign the AUXOUT: device to one serial printer, and assign the LST: device to the (default) parallel printer.

5.4 ALLOCATING CPU TIME TO THE FOREGROUND PROGRAM

Normally, the foreground program is allocated more time than a background program. The distributed system gives the foreground program 16 units of time and each background program 1 unit of time. The FORE command changes this ratio. For example, if you want to give the foreground 5 units of time and the background program 1 unit of time, enter the following:

```
A>fore 5

EOF
```

CPMPUG7.WS4 (= "CP/M-86 Plus User's Guide", section 7)

(Edited by Emmanuel ROCHE.)

Section 7: ED, The CP/M-86 Plus Context Editor

7.1 INTRODUCTION TO ED

To do almost anything with a computer, you need a way to enter data, to give the computer the information to process. The programs most commonly used for this task are called "editors". They transfer your keystrokes at the keyboard to a disk file. CP/M-86 Plus's editor is named ED. Using ED, you can easily create and alter CP/M-86 Plus text files.

The correct command format for invoking the CP/M-86 Plus editor is given in "Starting ED" of this section. After starting ED, you issue commands that transfer text from a disk file to memory for editing. "ED Operation" details this operation, and describes the basic text transfer commands that allow you to easily enter and exit the editor.

In this section, "Basic Editing Commands" details the commands that edit a file. "Combining ED Commands" describes how to combine the basic commands to edit more efficiently. Although you can edit any file with the basic ED commands, ED provides several more commands that perform more complicated editing functions, as described in "Advanced ED Commands"

During an editing session, ED can return two types of error messages. "ED Error Messages" lists these messages, and provides examples that indicate how to recover from common editing error conditions.

7.2 STARTING ED

Syntax: ED input-filespec {d: | output-filespec}

To start ED, enter its name after the CP/M-86 Plus prompt. The command ED must be followed by a file specification, one that contains no wildcard characters, such as:

```
A>ED MYFILE.TEX
```

The file specification, MYFILE.TEX in the preceding example, specifies a file to be edited or created. The file specification can be preceded by a drive specifier, but a drive specifier is unnecessary if the file to be edited is on your default drive. Optionally, the file specification can be followed by a drive specifier, as shown in the following example:

```
A>ED MYFILE.TEX B:
```

In response to this command, ED opens the file to be edited, MYFILE.TEX, on drive A, but sends all the edited material to a file on drive B.

Optionally, you can send the edited material to a file with a different filename, as in the following example:

```
A>ED MYFILE.TEX YOURFILE.TEX
```

The file with the different filename cannot already exist, or ED prints the following message and terminates:

```
Output File Exists, Erase It
```

The ED prompt ("*") appears at the screen when ED is ready to accept a command, as follows:

```
A>ED MYFILE.TEX
   : *
```

If no previous version of the file exists on the current disk, ED automatically creates a new file, and displays the following message:

```
NEW FILE
   : *
```

Note: Before starting an editing session, use the SHOW command to check the amount of free space on your disk. The unused portion of your disk must be at least as large as the file that you are editing, or larger if you plan to add characters to the file. When ED finds a disk or directory full, ED has only limited recovery mechanisms. These are explained in "ED Error Messages" later in this section.

7.3 ED OPERATION

With ED, you change portions of a file that pass through a memory buffer. When you start ED with one of the preceding commands, this memory buffer is empty. At your command, ED reads segments of the source file, for example MYFILE.TEX, into the memory buffer for you to edit. If the file is new, you must insert text into the file before you can edit. During the edit, ED writes the edited text onto a temporary work file, MYFILE.$$$.

When you end the edit, ED writes the memory buffer contents to the temporary file, followed by any remaining text in the source file. ED then changes the name of the source file from MYFILE.TEX to MYFILE.BAK, so you can reclaim this original material from the back-up file, if necessary. ED then renames the temporary file, MYFILE.$$$, to MYFILE.TEX, the new edited file.

Note: When you invoke ED with two filespecs, an input file and an output file, ED does not rename the input file to type BAK; therefore, the input file can be Read-Only or on a write-protected disk, if the output file is written to another disk.

The memory buffer is logically between the source file and the temporary work file. ED supports several commands that transfer lines of text between the source file, the memory buffer, and the temporary, and eventually final, file. The following table lists the three basic text transfer commands that allow you to easily enter the editor, write text to the temporary file, and exit the editor.

Table 7-1. Text Transfer Commands

Command	Result
nA	Append the next n unprocessed source lines from the source file to the end of the memory buffer.
nW	Write the first n lines of the memory buffer to the temporary file free space.
E	End the edit. Copy all buffered text to the temporary file, and copy all unprocessed source lines to the temporary file. Rename files.

7.3.1 Appending Text into the Buffer

When you start ED and the memory buffer is empty, you can use the A (append) command to add text to the memory buffer.

Note: ED can number lines of text to help you keep track of data in the memory buffer. The colon (":") that appears when you start ED indicates that line numbering is on. Type -V after the ED prompt to turn the line number display off. Line numbers appear on the screen, but never become a part of the output file.

The V (Verify Line Numbers) Command

The V command turns the line number display in front of each line of text on or off. The V command also displays the free bytes and total size of the memory buffer. The V command takes the following forms:

V, -V, 0V

Initially, the line number display is on. Use -V to turn it off. If the memory buffer is empty, or if the current line is at the end of the memory buffer, ED represents the line number as five blanks. The 0V command prints the memory buffer statistics in the form:

free/total

where free is the number of free bytes in the memory buffer, and total is the size of the memory buffer. For example, if you have a total of 48,253 bytes in the memory buffer and 46,652 of them are free, the 0V command displays this

information as follows

46652/48253

If the buffer is full, the first field, which indicates free space, is blank.

The A (Append) Command

The A command appends, or copies, lines from an existing source file into the memory buffer. The A command takes the following form:

nA

where n is the number of unprocessed source lines to append into the memory buffer. If a pound sign ("#") is given in place of n, then the integer 65,535 is assumed. Because the memory buffer can contain most reasonably sized source files, it is often possible to issue the command #A at the beginning of the edit, to read the entire source file into memory.

When n is 0, ED appends the unprocessed source lines into the memory buffer until the buffer is approximately half full. If you do not specify n, ED appends one line from the source file into the memory buffer.

7.3.2 ED Exit

You can use the W (Write) command and the E (Exit) command to save your editing changes. The W command writes lines from the memory buffer to the new file, without ending the ED session. An E command saves the contents of the buffer and any unprocessed material from the source file, and exits ED.

The W (Write) Command

The W command writes lines from the buffer to the new file. The W command takes the form:

nW

where n is the number of lines to be written from the beginning of the buffer to the end of the new file. If n is greater than 0, ED writes n lines from the beginning of the buffer to the end of the new file. If n is 0, ED writes lines until the buffer is half empty. The 0W command is a convenient way of making room in the memory buffer for more lines from the source file. If the buffer is full, you can use the 0W command to write half the contents of the memory buffer to the new file. You can use the #W command to write the entire contents of the buffer to the new file. Then, you can use the 0A command to read in more lines from the source file.

Note: After a W command is executed, you must enter the H command to reedit

the saved lines during the current editing session.

The E (Exit) Command

An E command performs a normal exit from ED. The E command takes the form:

```
E
```

followed by a ENTER.

When you enter an E command, ED first writes all data lines from the buffer and the original source file to the $$$ file. If a BAK file exists, ED deletes it, then renames the original file with the BAK filetype. Finally, ED renames the $$$ file from filename.$$$ to the original filetype, and returns control to the operating system.

The operation of the E command makes it unwise to edit a back-up file. When you edit a BAK file and exit with an E command, ED erases your original file because it has a BAK filetype. To avoid this, always rename a back-up file to some other filetype before editing it with ED.

Note: Any command that terminates an ED session must be the only command on the line.

7.4 BASIC EDITING COMMANDS

The text transfer commands discussed previously allow you to easily enter and exit the editor. This section discusses the basic commands that edit a file.

ED treats a file as a long chain of characters grouped in lines. ED displays and edits characters and lines in relation to an imaginary device called the character pointer (CP). During an edit session, you must mentally picture the CP's location in the memory buffer, and issue commands to move the CP and edit the file.

The following commands move the character pointer, or display text in the vicinity of the CP. These ED commands consist of a numeric argument and a single command letter, and must be followed by a ENTER. The numeric argument, n, determines the number of times ED executes a command; however, there are four special cases to consider in regard to the numeric argument:

1) If the numeric argument is omitted, ED assumes an argument of 1.

2) Use a negative number if the command is to be executed back through the memory buffer. The B command is an exception.

3) If you enter a pound sign ("#") in place of a number, ED uses the value 65,535 as the argument. A pound sign argument can be preceded by a minus sign ("-") to cause the command to execute back through the memory buffer ("-#").

4) 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 following table alphabetically summarizes the basic editing commands and their valid arguments.

Table 7-2. Basic Editing Commands

Command	Action
B, -B	Move CP to the beginning (B) or end (-B) of the memory buffer.
nC, -nC	Move CP n characters forward (nC) or back (-nC) through the memory buffer.
nD, -nD	Delete n characters before (-nD) or after (nD) the CP.
I	Enter insert mode.
IstringCtrl-Z	Insert a string of characters.
nK, -nK	Delete (kill) n lines before the CP (-nK) or after the CP (nK).
nL, -nL	Move the CP n lines forward (nL) or back (-nL) through the memory buffer.
nT, -nT	Type n lines before the CP (-nT) or after the CP (nT).
n, -n	Move the CP n lines before the CP (-n) or after the CP (n) and display the destination line.

The following sections discuss ED's basic editing commands in more detail. The examples in these sections illustrate how the commands affect the position of the character pointer in the memory buffer. Later examples in "Combining ED Commands" of this section illustrate how the commands appear at the screen. For these sections, however, the symbol _ in command examples represents the character pointer, which you must imagine in the memory buffer.

7.4.1 Moving the Character Pointer

This section describes commands that move the character pointer in useful increments, but do not display the destination line. Although ED is used primarily to create and edit program source files, the following sections present a simple text as an example to make ED easier to learn and understand.

The B (Beginning/Bottom) Command

The B command moves the CP to the beginning or bottom of the memory buffer. The B command takes the following forms:

B, -B

-B moves the CP to the end, or bottom, of the memory buffer; B moves the CP to the beginning of the buffer.

The C (Character) Command

The C command moves the CP forward or backward the specified number of characters. The C command takes the following forms:

nC, -nC

when n is the number of characters the CP is 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 beginning of the line and the top of the buffer. You can enter an n large enough to move the CP to a different line. However, each line is separated from the next by two invisible characters: a carriage return and a line-feed, represented by <cr><lf>. You must compensate for their presence. For example, if the CP is pointing to the beginning of the line, the command 30C moves the CP to the next line:

Emily Dickinson said, <cr><lf>
"I fin_d ecstasy in living - <cr><lf>

The L (Line) Command

The L command moves the CP the specified number of lines. After an L command, the CP always points to the beginning of a line. The L command takes the following forms:

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 back toward the beginning of the buffer. The command 2L moves the CP two lines forward through the memory buffer and positions the character pointer at the beginning of the line.

"I find ecstasy in living - <cr><lf>
the mere sense of living <cr><lf>
_is joy enough." <cr><lf>

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.

The n (Number) Command

The n command moves the CP, and displays the destination line. The n command takes the following forms:

n, -n

where n is the number of lines the CP is to be moved. In response to this command, ED moves the CP forward or backward the number of lines specified, then prints only the destination line. For example, the command -2 moves the CP back two lines.

```
 Emily Dickinson said, <cr><lf>
_"I find ecstasy in living - <cr><lf>
 the mere sense of living <cr><lf>
 is joy enough." <cr><lf>
```

A further abbreviation of this command is to enter no number at all. In response to a ENTER without a preceding command, ED assumes a n command of 1, and moves the CP down to the next line and prints it, as follows:

```
 Emily Dickinson said, <cr><lf>
 "I find ecstasy in living - <cr><lf>
_the mere sense of living <cr><lf>
```

Also, a minus sign ("-") without a number moves the CP back one line.

7.4.2 Displaying Memory Buffer Contents

ED does not display the contents of the memory buffer until you specify which part of the text you want to see. The T command displays text without moving the CP.

The T (Type) Command

The T command types a specified number of lines from the CP at the screen. The T command takes the forms:

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 after the CP. If no number is specified, ED types from the character pointer to the end of the line. The CP remains in its original position, no matter how many lines are typed. For example, if the character pointer is at the beginning of the memory buffer, and you instruct ED to type four lines (4T), four lines are displayed at the screen, but the CP stays at the beginning of

line 1.

_Emily Dickinson said, <cr><lf>
"I find ecstasy in living - <cr><lf>
the mere sense of living <cr><lf>
is joy enough." <cr><lf>

If the CP is between two characters in the middle of the line, a T command with no number specified types only the characters between the CP and the end of the line, but the character pointer stays in the same position, as shown in the following memory buffer example:

"I find ec_stasy in living -

When ED displays text with the T command, you can enter Ctrl-S to stop the display, then press any key when you are ready to continue scrolling. Enter Ctrl-C to abort long type-outs.

7.4.3 Deleting Characters

The D (Delete) Command

The D command deletes a specified number of characters, and takes the forms:

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. If the character pointer is positioned in the memory buffer as follows:

Emily Dickinson said, <cr><lf>
"I find ecstasy in living - <cr><lf>
the mere sense of living <cr><lf>
is joy _enough." <cr><lf>

the command 6D deletes the six characters after the CP, and the resulting memory buffer looks like this:

Emily Dickinson said, <cr><lf>
"I find ecstasy in living - <cr><lf>
the mere sense of living <cr><lf>
is joy _." <cr><lf>

You can also use a D command to delete <cr><lf> between two lines, to join them. Remember that the <cr> and <lf> are two characters.

The K (Kill) Command

The K command kills or deletes whole lines from the memory buffer, and takes the forms:

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. If the character pointer is at the beginning of the second line:

```
 Emily Dickinson said, <cr><lf>
_"I find ecstasy in living - <cr><lf>
 the mere sense of living <cr><lf>
 is joy enough." <cr><lf>
```

then the command -K deletes the previous line, and the memory buffer changes:

```
_"I find ecstasy in living - <cr><lf>
 the mere sense of living <cr><lf>
 is joy enough." <cr><lf>
```

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. A 0K command deletes the characters on the line up to the CP.

You can use the special # character to delete all the text from the CP to the beginning or end of the buffer. Be careful when using #K, because you cannot reclaim lines after they are removed from the memory buffer.

7.4.4 Inserting Characters into the Memory Buffer

The I (Insert) Command

To insert characters into the memory buffer from the console, use the I command. If you enter the command in uppercase, ED automatically converts the string to uppercase. The I command takes the forms:

I
Istring^Z

When you type the first command, ED enters insert mode. In this mode, all keystrokes are added directly to the memory buffer. ED enters characters in lines, and does not start a new line until you press the ENTER key.

```
A>ED B:QUOTE.TEX
  NEW FILE
     : *i
    1:  Emily Dickinson said,
```

```
2:  "I find ecstasy in living -
3:  the mere sense of living
4:  is joy enough."
5:  ^Z
 : *
```

Note: To exit from insert mode, you must press Ctrl-Z or ESC. When the ED prompt ("*") appears on the screen, ED is not in insert mode.

In command mode, you can use CP/M-86 Plus command line-editing control characters. In insert mode, you can use the control characters listed in the following table.

Table 7-3. CP/M-86 Plus Line-editing Controls

Command	Result
Ctrl-H	Delete the last character typed on the current line.
Ctrl-U	Delete the entire line currently being typed.
Ctrl-X	Delete the entire line currently being typed. Same as Ctrl-U.
<---	Remove the last character.

When entering a combination of numbers and letters, you might find it inconvenient to press a CAPS-LOCK key if your terminal translates the uppercase of numbers to special characters. ED provides two ways to translate your alphabetical 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, are translated to uppercase. If you enter the insert command letter in lowercase, all alphabetics are inserted as typed. The second method is to enter a U command before inserting text. Uppercase translation remains in effect until you enter a -U command.

The Istring^Z (Insert String) Command

The second form of the I command does not enter insert mode. It inserts the character string into the memory buffer, and returns immediately to the ED prompt. You can use CP/M-86 Plus's line-editing control characters to edit the command string.

To insert a string, first use one of the commands that position the CP. You must move the CP to the place where you want to insert a string. For example, if you want to insert a string at the beginning of the first line, use a B command to move the CP to the beginning of the buffer. With the CP positioned correctly, enter an insert string, as follows:

```
iIn 1870, ^Z
```

This inserts the phrase "In 1870," at the beginning of the first line, and returns immediately to the ED prompt. In the memory buffer, the CP appears after the inserted string, as follows:

    In 1870, _Emily Dickinson said, <cr><lf>

7.4.5 Replacing Characters

The S (Substitute) Command

The S command searches the memory buffer for the specified string but, when it finds it, automatically substitutes a new string for the search string. Whenever you enter a command in uppercase, ED automatically converts the string to uppercase. The S command takes the form:

    nSsearch string^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. For example, the command:

    sEmily Dickinson^ZThe poet

searches for the first occurrence of "Emily Dickinson" and substitutes "The poet." In the memory buffer, the CP appears after the substituted phrase, as follows:

    The poet_ said, <cr><lf>

If uppercase translation is enabled 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 Ctrl-Z.

7.5 COMBINING ED COMMANDS

You save keystrokes and editing time by combining the editing and display commands. You can type any number of ED commands on the same line. ED executes the command string only after you press ENTER. Use CP/M-86 Plus's line-editing controls to manipulate ED command strings.

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

1) The combined-command line must not exceed CP/M-86 Plus's 128-character maximum.

2) If the combined-command line contains a character string, the line

must not exceed 100 characters.

3) Commands to terminate an editing session must not appear in a combined-command line.

4) Commands that require character strings or filespecs such as the I, J, R, S, and X commands, must be either the last command on a line or must be terminated with a Ctrl-Z or ESC character, even if no character string or filespec is given.

While the examples in the previous section show the memory buffer and the position of the character pointer, the examples in this section show how the screen looks during an editing session. Remember that the character pointer is imaginary, but you must picture its location because ED's commands display and edit text in relation to the character pointer.

7.5.1 Moving the Character Pointer

To move the CP to the end of a line without calculating the number of characters, combine an L command with a C command, L-2C. This command string accounts for the <cr><lf> sequence at the end of the line.

Change the C command in this command string to move the CP more characters to the left. You can use this command string if you must make a change at the end of the line and you do not want to calculate the number of characters before the change, as in the following example:

```
1: *T
1:  Emily Dickinson said,
1: *L-7CT
   said,
1: *
```

7.5.2 Displaying Text

A T command types from the CP to the end of the line. To see the entire line, you can combine an L command and a T command. Type 0lt to move the CP from the middle to the beginning of the line, and then display the entire line. In the following example, the CP is in the middle of the line. 0L moves the CP to the beginning of the line. T types from the CP to the end of the line, allowing you to see the entire line. The command 0TT displays the entire line without moving the CP.

```
3: *T
   sense of living
3: *0LT
3:  the mere sense of living
3: *
```

To verify that an ED command moves the CP correctly, combine the command with

the T command to display the line. The following example combines a C command and a T command:

```
2: *8CT
   ecstasy in living -
2: *
4: *B#T
1:  Emily Dickinson said,
2:  "I find ecstasy in living -
3:  the mere sense of living
4:  is joy enough."
1: *
```

7.5.3 Editing

To edit text and verify corrections quickly, combine the edit commands with other ED commands that move the CP and display text. Command strings like the one that follows move the CP, delete specified characters, and verify changes quickly.

```
1: *15C5D0LT
1:  Emily Dickinson,
1: *
```

Combine the edit command K with other ED commands to delete entire lines and verify the correction quickly, as follows:

```
1: *2L2KB#T
1:  Emily Dickinson said,
2:  "I find ecstasy in living -
1: *
```

The abbreviated form of the I (insert) command makes simple textual changes. To make and verify these changes, combine the I command string with the C command and the 0LT command string, as follows. Remember that the insert string must be terminated by a Ctrl-Z.

```
1: *20Ci to a friend^Z0LT
1:  Emily Dickinson said to a friend,
1: *
```

7.6 ADVANCED ED COMMANDS

The basic editing commands discussed previously allow you to use ED for all your editing. The following ED commands, however, enhance ED's usefulness.

7.6.1 Moving the CP and Displaying Text

The P (Page) Command

Although you can display any amount of text at the screen with a T command, it is sometimes more convenient to page through the buffer, viewing whole screens of data and moving the CP to the top of each new screen at the same time, with ED's P command. The P command takes the following forms:

nP, -nP

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

To display the current page without moving the CP, enter 0P. The special character 0 prevents the movement of the CP. If you specify a negative number for n, P pages back towards the top of the file.

The n: (Line Number) Command

When line numbers are being displayed, ED accepts a line number as a command to specify a destination for the CP. The line number command takes the following form:

n:

where n is the number of the destination line. This command places the CP at the beginning of the specified line. For example, the command 4: moves the CP to the beginning of the fourth line.

Remember that ED dynamically renumbers text lines in the buffer each time a line is added or deleted. Therefore, the number of the destination line you have in mind can change during editing.

The :n (Through Line Number) Command

The inverse of the line number command specifies that a command should be executed through a certain line number. You can use this command with only three ED commands: the K (kill) command, the L (line) command, and the T (type) command. The :n command takes the following form:

:ncommand

where n is the line number through which the command is to be executed. The :n part of the command does not move the CP, but the command that follows it might.

You can combine n: with :n to specify a range of lines through which a command should be executed. For example, the command 2::4T types the second, third, and fourth lines:

```
1: *2::4T
2:  "I find ecstasy in living -
3:  the mere sense of living
4:  is joy enough."
2: *
```

7.6.2 Finding and Replacing Character Strings

ED supports a find command, F, that searches through the memory buffer, and places the CP after the word or phrase that you want. The N command allows ED to search through the entire source file, instead of just the buffer. The J command searches for and then juxtaposes character strings.

The F (Find) Command

The F command performs the simplest find function; it takes the form:

```
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. In the following example, the second occurrence of "living" is found.

```
1: *2fliving
3: *
```

The character pointer moves to the beginning of the third line where the second occurrence of "living" is located. To display the line, combine the find command with a type command. Note that, if you follow an F command with another ED command on the same line, you must terminate the string with a Ctrl-Z, as follows:

```
1: *2fliving^Z0lt
3: *the mere sense of living
```

It makes a difference whether you enter the F command in upper- or lowercase. If you enter F, ED internally translates the argument string to uppercase. If you specify f, ED looks for an exact match. For example, Fcp/m searches for CP/M but fcp/m searches for cp/m, and cannot find CP/M.

If ED does not find a match for the string in the memory buffer, it issues the message:

```
BREAK "#" AT
```

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

The N Command

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

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 has the same effect on an N command as it does on an F command. Note that, if you follow an N command with another ED command, you must terminate the string with a Ctrl-Z.

When an N command is executed, ED searches the memory buffer for the specified string, but, if ED does not find the string, it does not issue an error message. Instead, ED automatically writes the searched data from the buffer into the new file. Then, ED performs a 0A command to fill the buffer with unsearched data from the source file. ED continues to search the buffer, write out data, and append new data until it either finds the string, or reaches the end of the source file. If ED reaches the end of the source file, ED issues the following message:

BREAK "#" AT

Because ED writes the searched data to the new file before looking for more data in the source file, ED usually writes the contents of the buffer to the new file before finding the end of the source file and issuing the error message.

Note: You must use the H command to continue an edit session after the source file is exhausted and the memory buffer is emptied.

The J (Juxtapose) Command

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 the DELETE-TO STRING. This juxtaposes the string between the search and delete-to strings with the insert string. The J command takes the form:

nJsearch string^Zinsert string^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. In the following example, ED searches for the word "Dickinson", inserts the phrase "told a friend" after it, and then deletes everything up to the comma.

```
1: *#T
1:  Emily Dickinson said,
```

```
2:  "I find ecstasy in living -
3:  the mere of living
4:  is joy enough."
1: *jDickinson^Z told a friend^Z,
1: *0lt
1:  Emily Dickinson told a friend,
1: *
```

If you combine this command with other commands, you must terminate the delete-to string with a Ctrl-Z or ESC, as in the following example. If an uppercase J command letter is specified, ED looks for uppercase search and delete-to strings, and inserts an uppercase insert string.

The J command is especially useful when revising comments in assembly language source code, as follows:

```
236:  SORT    LXI    H, SW    ;ADDRESS TOGGLE SWITCH
236: *j;^ZADDRESS SWITCH TOGGLE^Z^L^Z0LT
236:  SORT    LXI    H, SW    ;ADDRESS SWITCH TOGGLE
236: *
```

In this example, ED searches for the first semicolon and inserts ADDRESS SWITCH TOGGLE after the mark and then deletes to the <cr><lf> sequence, represented by Ctrl-L. In any search string, you can use Ctrl-L to represent a <cr><lf> when the phrase that you want extends across a line break. You can also use a Ctrl-I in a search string to represent a tab.

Note: If long strings make your command longer than your screen line length, enter a Ctrl-E to cause a physical carriage return at the screen. A Ctrl-E returns the cursor to the left edge of the screen, but does not send the command line to ED. Remember that no ED command line containing strings can exceed 100 characters. When you finish your command, press ENTER to send the command to ED.

The M (Macro) Command

An ED macro command, M, can increase the usefulness of a string of commands. The M command allows you to group ED commands together for repeated execution. The M command takes the following form:

```
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 the source file, depending on the commands specified in the string. In the following example, ED executes the four commands repetitively until it reaches the end of the memory buffer:

```
1: *mfliving^Z-6diLiving^Z0lt
2:  "I find ecstasy in Living -
```

```
3:  the mere sense of Living

   BREAK "#" AT ^Z
3: *
```

The terminator for an M command is a ENTER; 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 Ctrl-Z or ESC. If a character string ends the combined-command string, it must be terminated by Ctrl-Z, then followed by a ENTER to end the M command.

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 source file. Usually, the command letter displayed in the message is one of the commands from the string, and not M.

To abort a macro command, press Ctrl-C.

The Z (Sleep) Command

Use the Z command to make the editor pause between operations. The pauses give you a chance to review your work. The Z command takes the following form:

```
nZ
```

where n is the number of seconds to wait before proceeding to the next instruction.

Usually, the Z command has no real effect unless you use it with a macro command. The following example shows you how you can use the Z command to cause a brief pause each time ED finds the word "living" in a file.

```
1: *mfliving^Z0tt10z
```

7.6.3 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 these lines from their original location, and finally an R command to read the block into its new location.

The X (Transfer) Command

The X command takes the forms:

```
nX
nXfilespec^Z
```

where n is the number of lines from the CP towards the bottom of the buffer that are to be transferred to a file. Therefore, n must always be a positive number. The nX command with no file specified creates a temporary file named X$$$$$$$.LIB. This file is erased when you terminate the edit session. The nX command with a file specified creates a file of the specified name. If no filetype is specified, LIB is assumed. This file is saved when you terminate the edit session. If the X command is not the last command on the line, the command must be terminated by a Ctrl-Z or ESC. In the following example, just one line is transferred to the temporary file:

```
1: *X
1: *t
1: *Emily Dickinson said,
1: *kt
1: *"I find ecstasy in living -
1: *
```

If no library file is specified, ED looks for a file named X$$$$$$$.LIB. If the file does not exist, ED creates it. If a previous X command already created the library file, 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 any file from within ED.

The R (Read) Command

The X command transfers the next n lines from the current line to a library file. The R command can retrieve the transferred lines. The R command takes the forms:

```
R
Rfilespec
```

If no filename is specified, X$$$$$$$ is assumed. If no filetype is specified, LIB is assumed. 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 is on a new line number. If you combine an R command with other commands, you must separate the filename from subsequent command letters with a Ctrl-Z as in the following example where ED types the entire file to verify the read.

```
1: *4l
 : *R^ZB#T
1:  "I find ecstasy in living -
2:  the mere sense of living
3:  is joy enough."
4:  Emily Dickinson said,
1: *
```

7.6.4 Saving or Abandoning Changes: ED Exit

You can save or abandon editing changes with the following three commands.

The H (Head of File) Command

An H command saves the contents of the memory buffer without ending the ED session, but it returns to the head of the file. It saves the current changes and lets you reedit the file without exiting ED. The H command takes the following form:

H

followed by a ENTER.

To execute an H command, ED first finalizes the new file, transferring all lines remaining in the buffer and the source file to the new file. Then ED closes the new file, erases any BAK file that has the same file specification as the original source file, and renames the original source file filename.BAK. ED then renames the new file, which has had the filetype $$$, with the original file specification. Finally, ED opens the newly renamed file as the new source file for a new edit, and opens a new $$$ file. When ED returns the * prompt, the CP is at the beginning of an empty memory buffer.

If you want to send the edited material to a file other than the original file, use the following command:

ED filespec differentfilespec

If you then restart the edit with the H command, ED renames the file differentfilename.$$$ to differentfilename.BAK, and creates a new file of differentfilespec when you finish editing.

The O (Original) Command

An O command abandons changes made since the beginning of the edit and allows you to return to the original source file and begin reediting without ending the ED session. The O command takes the form:

O

followed by a ENTER. When you enter an O command, ED confirms that you want to abandon your changes by asking the following:

O (Y/N)?

You must respond with either a Y or an N; if you press any other key, ED repeats the question. When you enter Y, ED erases the temporary file and the contents of the memory buffer. When the * prompt returns, the character pointer is pointing to the beginning of an empty memory buffer, just as it is

when you start ED.

The Q (Quit) Command

A Q command abandons changes made since the beginning of the ED session, and exits ED. The Q command takes the form:

```
Q
```

followed by a ENTER.

When you enter a Q command, ED verifies that you want to abandon the changes by asking the following:

```
Q (Y/N)?
```

You must respond with either a Y or an N; if you press any other key, ED repeats the question. When you enter Y, ED erases the temporary file, closes the source file, and returns control to CP/M-86 Plus.

Note: You can enter a Ctrl-BREAK or a Ctrl-C to return control immediately to CP/M-86 Plus. This does not give ED a chance to close the source or new files, but it prevents ED from deleting any temporary files.

7.7 ED ERROR MESSAGES

ED returns one of two types of error messages: an ED error message if ED cannot execute an edit command, or a CP/M-86 Plus error message if ED cannot read or write to the specified file. An ED error message takes the form:

```
BREAK "x" AT c
```

where x is one of the symbols defined in the following table and c is the command letter where the error occurred.

Table 7-4. ED Error Symbols

Symbol	Meaning
#	Search failure. ED cannot find the string specified in a F, S, or N command.
?c	Unrecognized command letter c. ED does not recognize the indicated command letter, or an E, H, O, or Q 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 anymore 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 keyboard aborted command execution.

F File error.
Followed by either DISK FULL or DIRECTORY FULL.

The following examples show how to recover from common editing error conditions. For example,

BREAK ">" AT A

means that ED filled the memory buffer before completing the execution of an A command. When this occurs, the character pointer is at the end of the buffer, and no editing is possible. Use the 0W command to write out half the buffer, or use an O or H command and reedit the 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 character pointer is at the end of the buffer. Move the CP with a B or n: line number command to resume editing.

BREAK "F" AT F
DISK FULL

Use the 0X command to erase an unnecessary file on the disk, or a B#Xd:buffer.sav command to write the contents of the memory buffer onto another disk.

BREAK "F" AT n
DIRECTORY FULL

Use the same commands described in the previous message to recover from this file error.

The following table defines the disk file error messages ED returns when it cannot read or write a file.

Table 7-5. ED Disk File Error Messages

Bdos Err on d: R/O
Function NNN File: FILENAME.TYP

Disk d: has Read-Only attribute. This error occurs if a different disk is inserted in the drive since the last cold or warm boot.

** FILE IS READ ONLY **

The file specified in the command to invoke ED has the R/O attribute. ED can

read the file so that the user can examine it, but ED cannot change a Read-Only file.

EOF

CPMPUGA.WS4 (= "CP/M-86 Plus User's Guide", appendix A)

(Edited by Emmanuel ROCHE.)

Appendix A: CP/M-86 Plus Control Character Summary

Table A-1. CP/M-86 Plus Control Characters

Char.	Meaning
RUB/DEL	Removes and echoes the last character if at the end of the line; otherwise, deletes the character to the left of the current cursor position.
Ctrl-A	Moves the cursor one character to the left.
Ctrl-B	Moves the cursor to the beginning of the command line, without having any effect on the contents of the line. If the cursor is at the beginning, Ctrl-B moves it to the end of the line.
Ctrl-C	Terminates the executing foreground program, and redisplays the system prompt.
Ctrl-E	Forces a physical Carriage Return on screen but does not send the command line to CP/M-86 Plus. Moves the cursor to the beginning of the next line without erasing the previous input.
Ctrl-F	Moves the cursor one character to the right.
Ctrl-G	Deletes the character indicated by the cursor. The cursor does not move.
Ctrl-H	Backspaces one character position when positioned at the end of the line; otherwise, deletes a character to the left of the cursor.
Ctrl-I	Moves the cursor to the next tab stop. Tab stops are automatically set at each eighth column. Has the same effect as pressing the TAB key.
Ctrl-J	Sends the command line to CP/M-86 Plus, and returns the cursor to the beginning of the line. Has the same effect as ENTER or Ctrl-M.
Ctrl-K	Deletes all characters to the right of the cursor, along with the character at the cursor.
Ctrl-M	Sends the command line to CP/M-86 Plus, and returns the cursor to the beginning of the line. Has the same effect as ENTER or Ctrl-J.
Ctrl-P	Echoes all console activity to the printer. A second Ctrl-P ends printer echo. Ctrl-P usually has no effect if your system does not include a printer, but this depends on your I/O system implementation.

Ctrl-Q Restarts screen scrolling after a Ctrl-S.

Ctrl-R Retypes the characters to the left of the cursor on the new line.

Ctrl-S Stops screen scrolling. If a display scrolls by too fast for you to read, type Ctrl-S. Type Ctrl-Q to restart screen scrolling.

Ctrl-U Discards all the characters in the command line, places a # at the current cursor position, and moves the cursor to the next line. However, you can use a Ctrl-W to recall any characters that were to the left of the cursor when you pressed Ctrl-U.

Ctrl-W Recalls and displays previously entered command line both at the operating system level and in executing programs, if the Ctrl-W is the first character entered after the prompt. If the command line contains characters, Ctrl-W moves the cursor to the end of the command line. If you press ENTER, CP/M-86 Plus executes the recalled command.

Ctrl-X Discards all the characters left of the cursor, and moves the cursor to the beginning of the current line. Ctrl-X saves any characters to the right of the cursor.

EOF

CPMPUGB.WS4 (= "CP/M-86 Plus User's Guide", appendix B)

(Edited by Emmanuel ROCHE.)

Appendix B: CP/M-86 Plus Filetypes

CP/M-86 Plus identifies every file by a unique file specification, which consists of a drive specifier, 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 usually indicates a special kind of file. The following table lists common filetypes and their meanings.

Table B-1. Common Filetypes

Type	Meaning
A86	Assembly language source file; ASM-86, the CP/M-86 Plus assembler, translates a file of type A86 into Intel 8086 machine language understood by the microprocessor of your computer.
ASC	File containing standard ASCII characters, that can be viewed on screen with the TYPE command, printed or sent to another computer, via the serial I/O ports, with the PIP command.
BAC	Console output file for background processes.
BAK	Back-up file created by a text editor; an editor renames the source file with this filetype to indicate that the original file has been processed. The original file stays on the disk as the back-up file, so you can refer to it.
BAS	Tokenized (not ASCII) source file for a BASIC interpreter.
CB1	CBASIC-86 pseudo-interpreter source file.
CB2	CBASIC-86 Compiler Version 2 (CB-86) source file.
CMD	8086 executable file.
H86	Program file in Intel hexadecimal format.
INT	CBASIC-86 intermediate language file.
LOG	Dr. Logo source file, in ASCII.
LST	Printable file that can be displayed on a console or printer.
SUB	Filetype required for SUBMIT input file, containing one or more CP/M-86 Plus commands. The SUBMIT program executes the commands in the file of type SUB, providing a batch mode for CP/M-86 Plus.

SYM Symbol table file.

WS4 WordStar Version 4.0 document file (8-bits).

$$$ Temporary file created by PIP, ED, and SUBMIT commands.

EOF

CPMPUGC.WS4 (= "CP/M-86 Plus User's Guide", appendix C)

(Edited by Emmanuel ROCHE.)

Appendix C: Glossary

application program:
Program that solves a specific problem. Typical applications programs are business accounting packages, word processing (editing) programs, and mailing list programs.

ASCII:
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.

attribute:
File or drive characteristic that can be set to ON or OFF.

background program:
Executing program that does not have access to the console, and shares computer resources with a foreground program.

back-up:
Copy of a disk or file made for safekeeping or the creation of the back-up disk or file.

bit:
A binary integer with a value of zero or one. A string of eight bits is used to represent numbers, letters, and symbols.

bootstrap:
Process of loading an operating system into memory. Bootstrap procedures vary from system to system. Typically, a miniloader is loaded automatically, and is 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.

byte:
Unit of memory or disk storage containing eight bits.

character string:
Any combination of letters, numbers, or special characters.

CCP:
Program (CCP.CMD) that produces the system prompt, allows you to change drives and user numbers, to change the allocated CPU time to the foreground program,

and contains the simple version (without options) of four other commands: DIR, ERASE, RENAME, and TYPE.

command:
Elements of a CP/M-86 Plus command line. Basically, a CP/M-86 Plus command has three parts: the command name, the command tail, and a ENTER keystroke.

command file:
Series of coded machine executable instructions stored on disk as a program file, invoked in CP/M-86 Plus by typing the command name next to the system prompt on the console. CP/M-86 Plus command files usually have a filetype of CMD. Files are either command files or data files. Same as a command program.

command name:
Name that identifies an CP/M-86 Plus command, usually the primary filename of a file of type CMD. The command name precedes the command tail and the ENTER in the command line.

command syntax:
Statement that defines the correct way to enter a command. A syntax line usually contains symbols that you replace with actual values when you enter the command.

command tail:
Part of a command that follows the command name in the command line. The command tail can include a drive specifier, a filename and/or filetype, a password, and options, but cannot exceed 128 characters. Some commands do not require a command tail.

concatenate:
Term that describes one of PIP's operations, that links together 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 or a printer, and a keyboard through which the user communicates with the operating system or applications program.

control character:
Combination of a Ctrl key and another character that sends a simple command to CP/M-86 Plus. Several control characters are recognized by CP/M-86 Plus and perform special functions, such as line editing. To enter a control character, press the Ctrl key on your terminal, and press the character key specified. See Appendix A.

cursor:
Usually, a blinking underscore or character cell that can appear anywhere on the console screen. The cursor indicates the position affected by the next keystroke at the console.

data file:
Non-executable collection of similar information, that usually requires a command file to manipulate it.

default:
System values used by CP/M-86 Plus commands and other programs when they are not specified in the command tail. System defaults include user, drive, or password. When the command A>DIR is entered, A is the default drive used by DIR when it displays filenames and types.

delimiter:
Special characters that separate different items in a command line. For example, in CP/M-86 Plus a colon (":") separates the drive specifier from the filename. A period (".") separates the filename from the filetype. Left and right square brackets ("[" and "]") separate any options from their command or filespec. Commas (",") separate one item in an option list from another. All of the preceding special characters are delimiters.

directory:
Portion of a disk that contains descriptions of each file on the disk. In response to the DIR command, CP/M-86 Plus displays the filenames stored in the directory.

DIR attribute:
File attribute. A file with the DIR attribute can be displayed by a DIR command. The file can be accessed from the default user number only.

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. 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. CP/M-86 Plus assigns a letter to each drive under its control. For example, CP/M-86 Plus can refer to the drives in a four-drive system as A, B, C, and D.

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 CP/M-86 Plus editor is invoked by typing the command ED next to the system prompt on the console. (See ED in Section 7.)

execute a program:
Transfer control of the computer to a program. When a program is running, the computer is executing a sequence of instructions.

file:
Collection of characters, instructions, or data stored on a disk, and identified in the disk directory by a filename and filetype.

File Control Block:
Structure used for accessing files on disk. Contains the drive, filename, filetype, and other information describing a file to be accessed or created on the disk.

filename:

First eight characters of a filename. The filename is a unique name that helps the user identify the file contents. A filename contains one to eight characters, and can include any letter or number, and some special characters.

file specification:
Unique file identifier. A complete CP/M-86 Plus file specification includes a disk drive specifier followed by a colon (d:), a primary filename of one to eight characters, a period ("."), a filetype of zero to three characters, a semicolon (";"), and a password of one to eight characters. For example, d:filename.typ;password is a complete CP/M-86 Plus file specification.

filetype:
Extension to a filename. A filetype can be from zero to three characters, and must be separated from the primary filename by a period ("."). A filetype should convey 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.

foreground program:
Program that is executing, and has control of the console. It can be the only executing program, or other programs can be executing in the background.

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.

input:
Data going into the system, usually from an operator typing at the terminal, or by a program reading from the disk.

I/O:
Abbreviation for input/output.

kilobyte:
1024 bytes denoted as 1K. 32 kilobytes are equal to 32K.

list device:
Device such as a printer onto which data can be listed or printed.

logical:
A name used to access a physical device, or a part of a physical device. The names for CP/M-86 Plus logical devices represent a class of physical devices. For example, LST: represents devices that can be used to print (display on paper) program output. The names A through P represent disk drives. A hard disk can be divided into several logical devices, and then accessed by the logical names.

megabyte:
Over one million bytes; 1024 kilobytes. See byte and kilobyte.

microprocessor:
Silicon chip that is the Central Processing Unit (CPU) of the microcomputer.

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.

option:
One of many parameters that can be part of a command tail. Use options to specify additional conditions for the execution of a command.

output:
Data that the system sends to the console or disk.

parameter:
Value in the command tail that provides information to the program (command).

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 with it.

physical:
Hardware of a computer. The physical environment varies from computer to computer.

program:
Series of specially coded instructions that perform specific tasks when executed by a computer.

program ID:
Unique number, that is created by CP/M-86 Plus, used to identify a running program.

program name:
Name of a running program, or a program file.

prompt:
Characters displayed on the screen when the system or a program requires a response from the user. 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 CP/M-86 Plus system prompt is a number and a letter followed by an angle bracket, such as 3A>. The number represents the current user number (except for zero, which is not displayed.) The letter indicates the current or default drive. Some applications programs have their own special prompts.

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 to change 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, or renaming a file. The SET command can set a file or a drive to Read-Only. Every file and drive is either Read-Only or Read/Write.

Read/Write:
Attribute that can be assigned to a disk file or a disk drive. If a file has the read/write attribute, you can read or write to this file. If a drive is set to read/write, you can read or write to any file that has the read/write attribute.

record:
Collection of data. A file consists of one or more records stored on disk. A CP/M-86 Plus record is 128 bytes long.

RO:
See Read-Only.

run a program:
See execute a program.

running program:
Executing program; see execute a program.

RW:
See Read/Write.

software:
Programs needed to make computers perform tasks, in contrast to hardware, which are the actual physical components of a computer.

source file:
Text file that is an input file for a processing program, such as an editor, text formatter, or assembler.

string:
See character string.

syntax:
Format for entering a given command.

system attribute:
File attribute. You can give a file the system attribute by using the SYS option in the SET command. A file with the SYS attribute is not displayed in response to a DIR command; you must use DIRSYS. If you give a file with user number 0 the SYS attribute, you can read and execute that file from any user number on the same drive.

system prompt:
Symbol displayed by the operating system, indicating that the system is ready to receive input. See prompt.

terminal:

See console.

user number:
Number from 0 to 15 assigned to a file when it is created. User numbers can organize files into 16 file groups.

utility:
Program that enables the user to perform certain operations, such as copying, erasing, or editing files. Utilities are created for the convenience of programmers and users.

wildcard characters:
Special characters that give CP/M-86 Plus a pattern to match when it searches the directory for a file. CP/M-86 Plus recognizes two wildcard characters, ? and *. The ? can be substituted for any single character in a filespec, and the * can be substituted for the filename or the filetype, or both. By placing wildcard characters in a filespec, you create a wildcard filespec, and then can quickly reference one or more files.

wildcard filename:
Filename that contains either of the CP/M-86 Plus wildcard characters, ? or *, in the primary filename or the filetype, or both. When you use wildcard characters, you create a wildcard filespec, and can easily reference more than one CP/M-86 Plus file. See Section 2 of this manual.

EOF

CPMPUGD.WS4 (= "CP/M-86 Plus User's Guide", appendix D)

(Edited by Emmanuel ROCHE.)

Appendix D: CP/M-86 Plus Error Handling

Table D-1. CP/M-86 Plus Error Messages

Format: Message
PROGRAM. Meaning

BREAK "x" AT c
ED. Where x is one of the following symbols, and c is the command letter being executed when the error occurs.

#
Search failure.
ED cannot find the string specified in an F, S, or N command.

?c
Unrecognized command letter c.
ED does not recognize the indicated command letter, or an E, H, Q, or O command is not alone on its command line.

O
The file specified in an R command cannot be found.

>
Buffer full.
ED cannot put any more characters in the memory buffer, or the string specified in an F, N, or S command is too long.

E
Command aborted.
A keystroke at the console aborted command execution.

F
Disk or directory full.
This error is followed by either the disk or directory full message. Refer to the recovery procedures listed under these messages.

Cannot edit wildcard filenames
ED. Ed cannot locate a file with an asterisk ("*") or a question mark ("?") file specification (wildcard). Reenter the command with the exact specification of the file that you want to edit.

Cannot find program ID: Program ID
TASK. The program ID that you entered is incorrect, or the program is already finished. Enter the TASK command or SHOW [PROGRAM] to display the names and ID's of the programs currently running.

Cannot find program name: program name
TASK. The program name that you entered is incorrect, or the program has already finished. Enter the TASK command or SHOW [PROGRAM] to display the names and ID's of the programs currently running.

Cannot have both create and access date/time stamps
SET. You cannot choose both create and access date/time stamps. You must select one or the other. Refer to the section in this "User's Guide" on the SET command for a description of the date/time stamping options.

Cannot have both page and no page options
No page is selected
SETDEF. You can choose only one of these options. If you try to select both, SET chooses the nopage option. If you want paging, you must reenter the SET command line with the page option only.

Cannot have file set to both RO and RW
SET. You can choose only one of these options. If you try to select both, SET returns this error message, and does not change the file attribute. Choose one or the other, and reenter the SET command line.

Cannot have file set to both SYS and DIR
SET. You can choose only one of these options. If you try to select both, SET returns this error message, and does not change the file attribute. Choose one or the other, and reenter the SET command line.

Cannot load CCP d>
CP/M-86 Plus. This prompt appears when CP/M-86 Plus cannot find the file CCP.CMD, when there is no space in memory to load the CCP program, or when a disk read/write error occurs while CP/M-86 Plus is loading the CCP program. When this prompt appears, the only things you can do are load another program, change drives, or change user numbers.

If you are working with application programs, you might not need the CCP. Place the application disk in the drive, and enter the application program command.

If the problem is caused by a disk read/write error, you probably received a message from CP/M-86 Plus (a message with "BDOS Function = xx" in the second line) at the same time as the "Cannot load CCP d>" prompt appeared. Refer to the description of that message in this appendix, or to the "Disk Read/Write Errors Checklist" at the end of this appendix.

If you did not receive any other message, the problem is either CP/M-86 Plus cannot find the CCP, or there is not enough space in memory to load the program. First, check your disks to locate CCP.CMD (CCP.CMD is usually on the system disk). Insert the correct disk, and enter CCP. You should see the CP/M-86 Plus system prompt.

If the disk with the CCP is in the correct drive, and you received the "Cannot load CCP d>" prompt, there is not enough memory available. If you have several programs running in the background, wait until they are finished, and try again. You might be able to load the TASK program, to check if your background

tasks have finished. If not, you can wait until either entering CCP or entering a Ctrl-C produces the CP/M-86 Plus system prompt.

Cannot load program
BACK, CP/M-86 Plus. This prompt appears when BACK or CP/M-86 Plus cannot load a program. Either there is no space in memory, or a disk read/write error occurred while loading the program.

If the problem was caused by a disk read/write error, you probably received a message from CP/M-86 Plus (a message with "BDOS Function = xx" in the second line) at the same time as the "Cannot load program" prompt appeared. Refer to the description of that message in this appendix, or to the "Disk Read/Write Errors Checklist" at the end of this appendix.

If you did not receive any other message, the problem is that there is not enough space in memory to load the program. If you have several programs running in the background, wait until they are finished, and try again. You might be able to load the TASK program, to check whether your background tasks have finished. If you do not have any other programs running, you do not have enough memory to run the program. Contact your dealer to see if it is possible to add more memory boards to your particular system configuration.

Command needs no further parameters
Extra parameters ignored
GET, PUT, SETDEF. The utility expects a shorter command line. Any further characters on the line are ignored. Refer to Section 6 for the specific command and a description of the command syntax.

CP/M Error on d: Disk Read/Write Error
BDOS Function = xx File = filespec
CP/M-86 Plus. This error message indicates a damaged disk or other disk read/write error. Refer to the "Disk Read/Write Error Checklist" at the end of this appendix for a list of possible disk read/write errors and their solutions.

CP/M Error on d: File Exists
BDOS Function = xx File = filespec
CP/M-86 Plus. You attempted to give a file a filename that already exists on that disk. Select another name or erase the old file, if you no longer need it.

CP/M Error on d: Invalid Drive
BDOS Function = xx File = filespec
CP/M-86 Plus. CP/M-86 Plus displays the preceding message when there is no disk in the drive, the drive door is open, or the power is OFF. It also displays this message when the drive selected is not supported by your system configuration. CP/M-86 Plus can support up to 16 drives, lettered A through P. Check the documentation provided by the manufacturer to find out which drives your particular system configuration supports. Specify the correct drive code, and reenter the command line.

CP/M Error on d: Password Error
BDOS Function = xx File = filespec
CP/M-86 Plus. The specified password is incorrect or invalid.

CP/M Error on d: ? in Filename
BDOS Function = xx File = filespec
CP/M-86 Plus. You used a wildcard file specification where it is not allowed. Refer to Section 6 for the specific command with the exact command syntax.

CP/M Error on d: Read-Only Disk
BDOS Function = xx File = filespec
CP/M-86 Plus. CP/M-86 Plus does not allow you to erase, rename, or update a file on a RO (Read-Only) drive. Use the SET command, SET d:[RW], to set the drive to RW (Read/Write), then reenter the command line.

CP/M Error on d: Read-Only File
BDOS Function = xx File = filespec
CP/M-86 Plus. CP/M-86 Plus does not allow you to edit, erase, rename, or update a file that is Read-Only (RO). Use the SET command, SET d:filespec[RW], to set the file to Read/Write (RW), then reenter the command line.

Date/Time Stamping Inactive
DIR. The DATE option is specified, but date/time stamping is not turned on for the disk directory. Refer to Section 6, "The SET Command," for instructions to turn on date/time stamping.

Destination is RO, Delete (Y/N)?
PIP. The destination file specified in a PIP command already exists, and it is set to RO (Read-Only). If you type Y (for Yes), the existing file is deleted and replaced with the file specified in the PIP command. If you do not want to delete the existing file, type N (for No) and reenter the PIP command with a different destination file specification: either specify a new filename, or direct the destination file to a different disk. This message can be avoided by using the W option. Refer to Section 6, "The PIP Command," for a discussion and list of the PIP options.

Directory full - d:filespec
BACK, ED, PIP, SUBMIT. There is not enough directory space for the output file. Use the ERASE command to remove unnecessary files, if any, from the full disk, or replace the disk before you reenter the command line. If you have another drive available, you can also redirect the output of the command to the disk on the other drive.

SUBMIT creates a temporary file from the SUB file it uses as input. Therefore, you must have at least one free directory entry to run SUBMIT.

If this error occurs while using ED, you can use the 0Xfilespec command to erase unnecessary files without leaving the editor. Alternatively, you can save the contents of the memory buffer on another disk with the command B#Xd:filespec, where d:filespec is a file on a different drive. You can then quit the edit. If you reedit the file, place the output on a different drive with the command ED filespec d:, where d: is a valid drive name other than the drive containing the source file. You can read the file saved with the Rd:filespec command.

Note: Part of the file might not be in the memory buffer when you save it (if you have not appended the whole file, or if you have issued any W commands).

Refer to Section 7 for further information on the ED command.

Directory needs to be reformatted for date/time stamps
SET. A date/time option is specified, but the directory is not initialized for date/time stamping. Refer to Section 6, "The INITDIR Command."

Disk full - d:filespec
BACK, ED, PIP, SUBMIT. The disk to which you are writing is full. Use the ERA command to erase unnecessary files, if any, from the full disk, or replace the disk before you reenter the command line.

SUBMIT creates a temporary file as large as the SUB file it is reading. Therefore, you must have as much empty space on your disk as the SUB file occupies.

This error might occur under ED on the W, E, H, or X commands. If it occurs with the X command, you can repeat the command, prefixing the filename with a different drive. Otherwise, you can try the recovery methods described above under the Directory Full error.

Disk Read/Write Error: d:filespec
HELP, PIP, RENAME, SUBMIT, TYPE. This error message indicates a damaged disk or other disk read/write error. Refer to the "Disk Read/Write Error Checklist" at the end of this appendix for a list of possible disk read/write errors and their solutions.

Disk read/write error, directory full, or HELP.HLP is read-only
Disk read/write error, directory full, or HELP.DAT is read-only
HELP. Use DIR [FULL] to check whether all the available directory entries have been used, or whether the file is Read-Only. If the directory is full, use the PIP command to copy the HELP files to another disk. When you finish working with the HELP files on the new disk, erase the HELP files from the original disk. Use PIP to replace HELP.HLP and HELP.CMD on the original disk.

If the files are Read-Only (RO), use the SET command, SET d:filespec[rw], to set them to Read/Write (RW). When you finish working with the HELP files, don't forget to use SET again to set the new HELP files to Read-Only.

If the directory is not full, and the HELP file indicated in the message is not Read-Only, refer to the "Disk Read/Write Errors Checklist" at the end of this appendix.

Disk read/write error, directory full, or no disk in drive: HELP.DAT
Disk read/write error, directory full, or no disk in drive: HELP.HLP
HELP. If there is a disk in the drive, use DIR [FULL] to check if all the available directory space on the disk is full. If the directory is full, use the PIP command to copy the HELP files to another disk. When you are finished working with them, erase the HELP files on the original disk, and use PIP to replace them from the new disk. If there is space in the directory, refer to the "Disk Read/Write Errors Checklist" at the end of this appendix.

Disk read/write error, disk full, or no disk in drive: HELP.DAT
Disk read/write error, disk full, or no disk in drive: HELP.HLP
HELP. If there is a disk in the drive, use DIR [FULL] to check if all the

available disk space is full. If the disk is full, use the PIP command to copy the HELP files to another disk. When you are finished working with the HELP files, erase them on the original disk, and use PIP to replace them from the new disk. If there is space on the disk, refer to the "Disk Read/Write Errors Checklist" at the end of this appendix.

Drive defined twice in search path
Delete one reference
SETDEF. A drive can be specified by name only once when you establish the search path order.

Drive specified in DRIVE option conflicts with drive in filespec
Select one
DIR. This error occurs when you specify different drives in the filespec and the DIR option. For example,

```
DIR a:erase.cmd[drive=b]
```

Omit one of the drive specifications. If you have two files of the same name on two drives and want to see both, try DIR erase.cmd[drive=(a,b)] or DIR erase.cmd[drive=all].

Error at end of line; nnn
GET, PUT, SETDEF. There is an input error at the end of the line indicated by nnn. Check your input file, and correct the error before reentering the command. Refer to the specific utility in Section 6 for more information on correct syntax and usage.

File already exists
Delete it? (Y/N)
RENAME. You ask CP/M-86 Plus to rename a file using a file specification that is already assigned to another file. Either delete the existing file, or use another file specification. If you enter Y, the existing file is deleted. If you enter N, nothing is done and CP/M-86 Plus returns the system prompt.

File contains an unprintable character at line NNN
SUBMIT. The SUBMIT file contains an invalid character in the line indicated. Use ED or your text editor to examine the line, and correct it before reentering the SUBMIT command. You might not be able to see the invalid character on your screen. The safest approach is to delete the entire line and retype it. This message can indicate that your SUBMIT file is corrupted. Refer to the section on "Corrupted Files" at the end of this appendix for information on recovery procedures for corrupted files.

File is empty
DUMP. Only a directory entry exists for the file.

File must have a password before you can set protection mode
SET. Use SET to assign a password to the file. Refer to "The Set Command," Section 6, or enter HELP SET.

File not found: filespec
BACK, DIR, DUMP, ED, ERASE, GET, HELP, PATCH, RENAME, SET, SUBMIT, TYPE.
A file that you specify cannot be found, or does not exist. Check that you

entered the correct drive specification, and that you have the correct disk in the drive. Refer to Section 4, "How CP/M-86 Plus Searches for Command and Data Files," for more information on the search path CP/M-86 Plus uses to find files.

This error can also occur when using the PIP command, if the file that you are trying to PIP is set to the SYS (system) attribute. Use the R option to PIP this file. PIP d:filespec = d:filespec[R].

If this error occurs with DIR, ERASE, REN, or TYPE, note that some of the internal CP/M-86 Plus commands have options that require support from a command file. The command file has the same name as the internal command, with a filetype of CMD. The file must be in your drive search chain, described in "The SETDEF Command," in Section 6.

File Password Error - d:filespec
DUMP, PATCH, PIP, RENAME, SET, TYPE.
The specified password is incorrect or invalid.

File Verification Failure - d:filespec
PIP. When copying with the V option, PIP finds a difference when rereading the data just written and comparing it to the data in its memory buffer. Usually, this error message indicates a disk read/write error. Refer to the "Disk Read/Write Error Checklist" at the end of this appendix for a list of possible disk errors and their solutions.

GET and PUT do not support AUXIN and AUXOUT
GET and PUT. GET and PUT do not support AUXIN and AUXOUT. Refer to the PIP and DEVICE commands in Section 6 for information on transferring data to and from auxiliary devices.

Group all options within one set of brackets
DIR. Reenter the command line with all DIR options inside one set of square brackets, separated by commas (",") or spaces. If you have two or more filespecs and want to use different options for each, enter separate DIR commands. You can do this on the same line by separating the DIR commands (with their command tails) by a !. For example, DIR filespec1 [option]! DIR filespec2 [option].

HELP.HLP must be on the default drive
HELP. The HELP.HLP file must be on the current, or default, drive. Copy it to your default drive from your CP/M-86 Plus system disk. This error can also occur when you try to access HELP from a user number other than 0. Because HELP.HLP is a data file, it must be set to the SYS attribute in user 0 to be accessible from other user numbers on that drive. (See Section 4 for more information.)

Invalid Command for Sparse File - d:filespec
PIP. This error usually occurs when you use PIP to concatenate or otherwise to change the format of a sparse file. Refer to the "CP/M-86 Plus Programmer's Guide" for more information on sparse files and their requirements.

Invalid delimiter in command line
BACK, GET, PIP, PUT, SETDEF, TYPE.

The delimiter [ ], =, or space, is not valid at the location used. For example, a [is used where an = is needed.

If this error occurs with PIP, you placed an invalid character for a separator between two input filenames. Reenter the PIP command line using a comma (",") as the separator between the input filenames.

Invalid destination device
PIP. The destination specified in your PIP command is not a valid PIP destination. You specified an input device as a destination, or used a wildcard file specification. Try the PIP command again, giving an output device or a specific file as the destination. Refer to "The PIP Command" in Section 6 for a discussion of the valid PIP destinations.

Invalid drive: d:filespec
PIP, SET, SETDEF, SHOW, TYPE.
The disk that you specified in the command line is not supported by your system. CP/M-86 Plus can support up to 16 drives, lettered A through P. Check the documentation provided by the manufacturer to find out which drives your system configuration supports. Specify the correct drive code, and reenter the command line. SETDEF also recognizes an at sign ("@") as the default drive.

Invalid filespec: d:filespec
Use d:filename.typ;password
ED, ERASE, GET, PIP, PUT, RENAME, SET, SUBMIT, TYPE.
The filename typed does not conform to the usual CP/M-86 Plus file naming conventions. The filename is too long, or a delimiter is used as the first character. Filenames cannot be longer than eight characters. Refer to "Naming Files" in Section 2 for a list and description of the delimiters that cannot be used as the first character in a filename. Correct the filename, and reenter the command line.

Invalid HEX Digit - d:filespec
PIP. An invalid hex digit is encountered while reading a hex file. This message indicates a truncated or corrupted hex file.

Invalid HEX File--Unexpected End - {filespec}
PIP. The last hex record is not as long as the character count indicated. The hex file is truncated. Refer to the section on "Corrupted Files" in the back of this appendix for more information on recovery procedures for corrupted files. (ROCHE> Missing...)

Invalid HEX Record--Checksum - d:filespec
PIP. A hex record checksum error is encountered during the transfer of a hex file. This message indicates a corrupted or truncated hex file. Refer to the section on "Corrupted Files" in the back of this appendix for more information on recovery procedures for corrupted files. (ROCHE> Missing...)

Invalid Hex Record--Too Long - d:filespec
PIP. A hex record exceeds 80 characters in a file being copied with the [H] or [I] option.

Invalid Source Device
PIP. The source specified in your PIP command is not a valid PIP source. You

specified an output device as a source, or used a wildcard filename. Reenter the PIP command line with an input device or a specific filename as the source. Refer to "The PIP Command" in Section 6 for a discussion of the legal sources for a PIP command.

Invalid User Number--Not in the range 0 - 15
PIP, USER. You specified a user number greater than 15. User numbers are from 0 to 15.

Not enough available memory
ED, HELP, INITDIR.
There is not enough memory available in your system configuration to run the program. Refer to the manufacturer's documentation, or contact the place where you purchased your system for assistance. With DIR, you can try the no-sort option; if you eliminate sorting, there might be enough room in memory to read the disk directory.

If you receive this message when using SUBMIT, you might have too many nested SUBMIT files for the buffer space available. Rewrite your SUB file so that it uses not more than eight levels of nesting.

Option required
SET. You must enter an option with the SET command. Refer to "The SET Command" in Section 6, or enter HELP SET for a listing and description of the SET options.

Option requires a filespec
SET. The SET option you specified must have a filespec entered with it. Refer to SET [HELP] or "The SET Command" in Section 6 for the correct syntax.

ORDER option requires filetype of CMD or SUB
SETDEF. The SETDEF order option can only be used for command (CMD) and submit (SUB) files. Refer to "The SETDEF Command" in Section 6.

Output file exists, erase it
ED. The destination filename already exists. Erase it, or select another disk to receive the output file.

PUT output file erased
PUT. PUT, when it finishes, tries to close the ouptput file it created. If the PUT output file is erased before PUT finishes, by a program or by you, this message appears.

Quit String Not Found
PIP. The string argument to a Q parameter is not found in your input file. Ensure that you type the correct string, and retry the operation. This error can also occur when entering the Q option from the command line. CP/M-86 Plus translates everything in the command line into uppercase. To avoid this problem, enter the Q option from inside PIP. Enter PIP, and wait for the * prompt before entering the file specifications and Q option.

Read-only disk: d:
PATCH, RENAME. CP/M-86 Plus does not allow you to erase, rename, or update a file on a RO (Read-Only) drive. Use the SET command, SET d:[RW], to set the

drive to RW (Read/Write), then reenter the command line.

This error also occurs when PIP cannot erase a temporary file, or create a new file on a drive that is set to RO. Use the recovery methods listed above, then reenter the PIP command.

Read-only file: d:filespec
ED, ERASE, PATCH, PUT, RENAME, SET.
CP/M-86 Plus does not allow you to edit, erase, rename, or update a file that is Read-Only (RO). Use the SET command, SET d:filespec[RW], to set the file to Read/Write (RW), then reenter the command line.

Requires CP/M-86 Plus to run
DATE, DEVICE, DIR, DUMP, ED, ERASE, GET, INITDIR, PIP, PUT, RENAME, SET, SETDEF, SHOW, SUBMIT, TYPE, TASK.
This version of the utility must only be run under CP/M-86 Plus.

Start String Not Found
PIP. The string argument to an S parameter cannot be found in the source file. Ensure that you are typing the correct string, and try the operation again. This error can also occur when entering the S option from the command line. CP/M-86 Plus translates everything in the command line into uppercase. To avoid this problem, enter the S option from inside PIP. Type PIP, and wait for the * prompt before typing the file specifications and S option string.

The specified option applies only to drives
SET. Refer to "The SET Command" in Section 6, or enter HELP SET for a discussion and listing of the SET options.

There is not enough remaining directory space to allow for the date/time extension.
INITDIR. Refer to the INITDIR utility in Section 6. Briefly, INITDIR requires that at least one third of the total directory space on a disk be available for date/time extensions. You can use the DIR [FULL] command to see how many of the available directory entries are in use on a disk.

This program cannot run when another program is running in the background, or when RSXs are present.
DEVICE, INITDIR. To run these programs, you must wait until the background programs are finished, or you can stop the background programs.

Too many entries in HELP file
HELP. There are too many entries in the Help index list for the amount of buffer space available. You can have no more than 256 entries in your HELP file. Eliminate some entries before reentering the HELP [CREATE] command.

Too many programs. Only three background programs allowed.
BACK. You can only run three background programs at a time. Wait until one of the other programs is finished before attempting to run the additional program. Use the TASK command to check if the background programs are finished.

Unable to close file - d:filespec
PIP. This message usually indicates a damaged disk. Ensure that the disk is

in place, and try the operation again. If the error persists, the disk is damaged and must be replaced. Refer to the instructions provided by your system's manufacturer on recovering damaged disks.

Unrecognized command line input
ERASE, GET, PIP, PUT, SET, SETDEF.
Usually a needed delimiter is omitted, or the format of your command line is incorrect. Refer to the specific command in Section 6, or enter HELP COMMAND for a description of the correct command syntax.

Unrecognized date/time specification
Use MM/DD/YY and HH:MM:SS
DATE. Refer to "The DATE Command" in Section 6, or enter HELP DATE for more information.

Unrecognized input at the "^": command line
GET, PUT, SETDEF.
The portion of the command line underscored by the "^" is incorrect. Refer to the specific command in Section 6, or enter HELP COMMAND for a description of the correct command syntax.

Unrecognized parameter on line nnn
Valid parameters are dollar sign (0-9)
SUBMIT. If you have a $ in your file that is not intended as a SUBMIT parameter, replace it with $$, which is read as a single $, and not mistaken for a parameter.

Unrecognized parameter or option
BACK, DIR, ERASE, GET, PIP, PUT, SETDEF, SHOW.
Either an option, modifier, or delimiter does not exist for this command, a necessary delimiter is missing, or an option typed in the command line is not valid for the command. Refer to the specific command in Section 6, or enter HELP COMMAND for a description of the correct command syntax.

Valid options are PAGE or NOPAGE
TYPE. TYPE has only two options, PAGE and NOPAGE. If you want to control the format of a file sent to the console, read "The PIP Command" in Section 6. Try PIPing the file to CON: with the options available to PIP.

Warning: Extra input ignored
SUBMIT. The SUBMIT file contains a line with <, indicating progam input, but the program does not require additional input. The line is ignored.

Wildcard in old file must match wildcard in new file
RENAME. The wildcard must appear in the same part of the filename or filetype in both the source and the destination.

You cannot specify more than four drives in the search chain
SETDEF. Refer to "The SETDEF Command" in Section 6 for a description of the SETDEF options.

Common Error Handling Procedures

This section discusses several classes of errors, their causes, and some recovery procedures.

Disk Read/Write Errors Checklist

Use the following checklist to help locate the cause of a disk read/write error.

* Is the disk correctly inserted?

  - If there is no disk in the drive, you might receive a "Disk Read/Write Error" message. Ensure that you have a disk in the drive before you continue.

  - If the disk is not securely in place, or if the drive door is open, you might also receive a disk read/write error. Ensure that the disk is correctly inserted, and that the drive door is securely closed, before you continue.

* Is the disk the correct format for the drive?

  - Some systems allow you to format your disks as either single or double density. If you place a double-density disk in a drive that accepts only single density, or vice versa, you might receive disk read-write errors. Refer to the documentation provided by your system's manufacturer for more information on formatting your disks and drives.

  - If you are using a single-sided disk in a drive set for double-sided disks, or a double-sided disk in a single-sided drive, you might receive errors. Check the manufacturer's documentation to ensure that your disks are correctly formatted for your system configuration.

* Is the disk physically write-protected? Your disk might be physically write-protected. Usually, a tab can be punched out, or you can cover a notch with a piece of tape to write-protect a disk. Typically, 5 1/4-inch diskettes are write-protected by the absence of a notch on the upper-right diskette edge, while 8-inch disks are write-protected by the presence of a notch on the bottom of the disk. Check the documentation provided by the disk manufacturer for instructions on changing the write-protection status of the disk.

* If the error is not caused by one of the preceding problems, the disk might be damaged. Refer to the instructions provided by your system's manufacturer on recovering damaged disks.

* If the error recurs, there could be a problem with your system.

  - Check for loose connections.

  - If the connections all seem tight, the drive might be damaged. Contact your dealer for assistance.

Reporting Potential Software Problems

If you have a problem that you cannot resolve, contact your dealer for assistance. Be prepared to provide the following information:

* Indicate the version of the operating system that you are using.

* Describe your system's hardware configuration: number of drives, quantity of memory, and additional equipment such as printers, modems, and so forth.

* Provide sufficient information to reproduce the error. Indicate the programs that were running at the time the error occurred. If possible, provide a disk with a copy of the program.

* Provide the text of the error message you received.

EOF