DIGITAL RESEARCH™

Concurrent CP/M-86™ Operating System User's Guide

Concurrent CP/M-86™ Operating System User's Guide

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

The <u>Concurrent CP/M-86 Operating System User's Guide</u> was prepared using the Digital Research TEX-80 Text Formatter and printed in the United States of America by Commercial Press/Monterey.

```
**********************************
*  First Edition:  August 1982  *
**********************************
```

Foreword

Welcome to the world of microcomputers opened to you by your IBM Personal Computer. Welcome also to the world of application software accessible through Digital Research Concurrent CP/M-86$^{T.M.}$. Digital Research designed this version of Concurrent CP/M-86 especially for the Intel® 8088 microprocessor that is the heart of your IBM Personal Computer.

Concurrent CP/M-86 lets you do two or more jobs with your computer at the same time. If you want to edit a paper (or two!) and also look at another, Concurrent CP/M-86 takes care of the details, lets you switch from one task to another with a touch of a key. You can even print one or more documents while you edit or review others. Concurrent CP/M-86 lets you make the most of your IBM Personal Computer and makes you more productive.

What Concurrent CP/M-86 Does For You

Concurrent CP/M-86 manages information stored magnetically on your diskettes by grouping this information into files of programs, which are sequences of computer instructions, and data, which is the raw material programs use to do work. Concurrent CP/M-86 can copy files from a diskette to the IBM Personal Computer's memory, or to a peripheral device such as a printer. Concurrent CP/M-86 performs these and other tasks by executing various programs according to commands you enter at your keyboard.

Once in memory, a program runs through a set of steps that instructs your computer to perform a certain task. The advantage of using Concurrent CP/M-86 on your IBM Personal Computer is that you can create your own Concurrent CP/M-86 programs to entertain, educate, or solve your own commercial or scientific problems.

What You Need To Run Concurrent CP/M-86 On The IBM Personal Computer

The minimum IBM Personal Computer consists of a System Unit with 16K of memory, a keyboard, and a screen device. However, Concurrent CP/M-86 needs some additional features to operate properly.

To start, Concurrent CP/M-86 needs at least two diskette drives. You can have up to four diskette drives. You also need at least 256K bytes of memory.

If you expand your system beyond these minimum requirements, you will appreciate that Concurrent CP/M-86 support many other features that can be added to your IBM Personal Computer. For example, Concurrent CP/M-86 supports the maximum amount of memory you can install--544K bytes. Concurrent CP/M-86 supports both low- and high-resolution color displays, and monochrome displays.

How This Book Is Organized

This book introduces you to Concurrent CP/M-86 and tells you how to use it. The book assumes you have read the Guide to Operations that accompanied your IBM Personal Computer and are familiar with the parts of your computer, how to set it up and turn it on, and how to handle, insert, and store diskettes. However, it does not assume you have had a great deal of experience with computers.

Section 1 tells how to start Concurrent CP/M-86 and enter commands. Section 2 discusses diskettes, files, diskette drives, and other devices associated with your IBM Personal Computer. Section 3 develops the Concurrent CP/M-86 command concepts you need to understand the command summary in Section 4. Section 4 lists all commands in alphabetical order, describing every command and program supplied with Concurrent CP/M-86. Section 5 tells you how to use ED, the Concurrent CP/M-86 file editor. With ED you can create and edit program, text, and data files.

When you become more familiar with your IBM Personal Computer's 8088 microprocessor instructions and you decide to write assembly language programs, will find that ASM-86 T.M., the Digital Research Assembler, simplifies the process. The companion program to ASM-86 is DDT-86 T.M., the Dynamic Debugging Tool. DDT-86 makes it easy to track down and eliminate errors in your assembly language programs. The companion manual of the Concurrent CP/M-86 Operating System User's Guide, the Concurrent CP/M-86 Operating System Programmer's Guide, describes ASM-86 and DDT-86 in detail.

Appendix A lists the error messages Concurrent CP/M-86 displays when it encounters special conditions. If the condition requires correction, Appendix A can also tell you what actions you should take before you proceed.

Appendix B is a glossary of computer terms that explains the concepts you encounter in learning about your Personal Computer.

Appendix C summarizes Concurrent CP/M-86 commands, and Appendix D summarizes Concurrent CP/M-86 control characters. Appendix D also describes the way some Concurrent CP/M-86 control keys differ from their counterparts in CP/M-86 T.M.. Appendix E describes the various Concurrent CP/M-86 filetypes.

Appendix F is a list of items to check if you have problems with your files. Appendix G summarizes the kinds of drive and file status information Concurrent CP/M-86 returns to you. Appendix H lists the console escape sequences.

If you are new to computers, you might find some of these topics confusing. Hang in there. Half the fun of an IBM Personal Computer is learning more about how to use it in your home, business, research, or school. This book proceeds step-by-step, so you can readily understand each Concurrent CP/M-86 operation.

Table of Contents

Table of Contents (continued)

Table of Contents (continued)

Table of Contents (continued)

Table of Contents (continued)

Appendixes

Section 1
Introduction to Concurrent CP/M-86

This section discusses the fundamentals of your IBM Personal Computer and its new operating system, Concurrent CP/M-86. It starts by describing Concurrent CP/M-86, tells you how to start it up, and shows what the initial messages look like on the screen. The remaining sections introduce the Concurrent CP/M-86 commands and line-editing control characters.

1.1 What Concurrent CP/M-86 Is

Concurrent CP/M-86 is a single-user, multitasking operating system for the IBM Personal Computer. It offers you a number of innovations that increase your productivity and make it easy--and fun--to get your work done.

1.1.1 Virtual Consoles

Concurrent CP/M-86 lets you execute several programs at one time (concurrently) by creating up to four virtual consoles. Virtual consoles are channels that you can switch back and forth between as you would switch channels on a television.

With a television, one channel might be broadcasting the news, another a sports event, and so on. With Concurrent CP/M-86, you control what you see on your terminal screen. On one channel you can write a letter, while another channel shows a directory of all your text files and other channels perform other tasks.

When a console is switched-in, appearing on your screen, we say that it is in the foreground. When a console is switched-out, invisible to you, we describe it as in the background.

You can switch from one virtual console to another by pressing the CTRL key while pressing 0, 1, 2, or 3 on the keypad. Press CTRL-0 to call Console 0, CTRL-1 for Console 1, CTRL-2 for Console 2, or CTRL-3 for Console 3.

1.1.2 Dynamic Mode

When a virtual console is in the Dynamic mode, all letters and numbers (characters) sent to it appear on the screen immediately. If you were editing a letter, for example, you would see the text of the letter on your screen and your changes would appear as you made them. If you switch-out a virtual console in the Dynamic Mode while characters are appearing on the screen, those characters are lost.

1.1.3 Buffered Mode

If a virtual console is switched out, none of the characters generated by a running program appear on your screen. In Buffered mode, Concurrent CP/M-86 creates storage for this information. All characters that go to the screen if the terminal is switched in are instead saved in a temporary file on a floppy diskette. When you switch in this virtual console the saved characters appear on your terminal screen. You do not lose a thing. You can let one program turn out pages of text while you work on another project, then return to see how the first program is doing.

Section 4.28, the VCMODE command, tells you how to use the VCMODE instruction to switch your consoles between Dynamic and Buffered Modes.

1.2 How To Get Concurrent CP/M-86 Started

Starting or loading Concurrent CP/M-86 means reading a copy of it from your boot diskette into your computer's memory. You can start Concurrent CP/M-86 in one of two ways, depending on whether your computer is powered on or off.

If power is off, insert your Concurrent CP/M-86 boot diskette with the label facing upward into drive A, the built-in drive on the left side of the System Unit, and place the Concurrent CP/M-86 command diskette in drive B, the built-in drive on the right side. Close the drive door. When you turn the power on, your IBM Personal Computer automatically loads CP/M-86 into memory after a few moments of self-testing. (If you just powered off but want to use your computer again, you should count slowly to five between turning the power off and then back on.)

1.2.1 System Reset

If power is on and you want to restart Concurrent CP/M-86, first make sure your Concurrent CP/M-86 diskettes are in their proper drives, hold down the CTRL and ALT keys, and press the DEL key. Release all three keys. When you press these keys in this way, the Personal Computer performs a System Reset. This means that any internal processing the Personal Computer is doing stops. The computer returns to the state it was in immediately after power-up; Concurrent CP/M-86 is then reloaded from the diskettes.

1.2.2 Sign-On Message

In either start sequence, after the Personal Computer completes its internal check, the light on drive A goes on, and the drive clicks and whirrs as Concurrent CP/M-86 load into memory. The first thing Concurrent CP/M-86 does after it is loaded into memory is display a sign-on message on your screen, like one in the following example:

```
    Hardware Supported:
                       Diskette(s)   : 2
         Parallel Printer Port(s)   : 1
           Serial Printer Port(s)   : 1
                 Main Memory (Kb)   : 320


    Concurrent CP/M-86 V.V [release date]
    Copyright (c) 1982 Digital Research
```

The release number, represented above by V.V, tells you the major and minor revision level of the Concurrent CP/M-86 version that you own.

1.2.3 Status Line

Concurrent CP/M-86 writes a status line at the bottom line of your screen. This status line communicates various important information about what Concurrent CP/M-86 is doing, which features are currently invoked, and where output from the Personal Computer will go.

Look at the status line from left to right as you read these descriptions.

The first message is "Console = n", where n is the current virtual console. Concurrent CP/M-86 comes up in virtual console 0. You can watch the console number change by pressing the CTRL key and numbers 0, 1, 2, or 3 on the keypad at the right of the keyboard.

The next message to the right is the mode for that virtual console (discussed in Section 1.1). As you switch from virtual console 0 to 1 to 2 to 3, you see the default mode message appear for each virtual console; they are all initially set to Dynamic. You can change the mode to Buffered with the VCMODE command described in section 4.28.

To the right of the background mode message is the Printer message. When you press CTRL-P, Concurrent CP/M-86 copies the contents of the screen to the current printer. The printer message tells the number of the current printer. The default printer is printer number 0, as you see, and you can change it with the PRINTER command (described in Section 4.17). Next to the printer message is a space for a ^P=n. The ^P=n appears when you press CTRL-P, telling you that output from the current virtual console is going to the printer and to your screen, and showing the printer number. Press CTRL-P again to turn printing off.

Next to the printer messages you see the process name message. This is the name of the process currently executing on your virtual console. If you just loaded Concurrent CP/M-86, you see "Tmp 0" here. This tells you that Concurrent CP/M-86 is currently running no user programs at this console, but the TMP

Processor) is looking at the keyboard, waiting for your commands (The Concurrent CP/M-86 Programmer's Guide has more details about the TMP and its activities).

To the right of the process name message appears the TOD (Time of Day) message. This is a 24-hour clock that indicates the current time in hh:mm:ss format. The TOD command sets this clock to a specified time. (See Section 4.25 for more about TOD.)

Next to the clock you see "Wrap". This tells you that the screen is set to wrap lines of text from the right edge of the screen over to the left if you type more than 80 characters without a carriage return, ↵ . If you disable Wrap by pressing ESC and w at the same time, Wrap disappears, and if you type more than 80 characters the balance are not displayed. You can turn Wrap back on with an ESC-v.

To the right of Wrap is "Caps", which indicates if you have pressed the Caps Lock key. This action sets the letter keys on the keyboard to all capital letters. Press it again to restore the keyboard to both upper- and lower-case.

Next to Caps is "Num", a message indicating what happens when you press the keypad keys. The Num Lock key switches the keypad keys back and forth; when Num appears in the Status Line, pressing a keypad key generates that number. Pressing the shift key, ⇧, and a keypad key generates the function specified for that key by the FUNCTION command (described in Section 4.13). If there's no Num in the status line, the reverse is true: pressing the shift key returns the number, and pressing the key alone generates the programmed function. Try pressing a few keys to become familiar with them.

To the right of Num is a space for a ^S. Sometimes Concurrent CP/M-86 reports text or data to you faster than the human eye can follow. When this happens, you can stop the text from rolling off the top of your screen (scrolling upwards) by pressing the CTRL key and the S key simultaneously, or by pressing the Scroll Lock key. This stops the screen in its tracks. When you press CTRL-S, the ^S indicator comes on, and none of the individual keys on the keyboard do anything but beep if you press them. To get going again, press CTRL and Q, or SHIFT-Scroll Lock.

The last message field, at the extreme right of the status line, shows if you have any open (currently being used) files on a particular drive. The characters that appear are A and B for these two diskette drives, and C and D if you have a four-drive system, M for the MDISK if you have implemented it, and an asterisk (*) for any further drives you have installed yourself. If you are new to Concurrent CP/M-86 and you have not encountered the expressions file and MDISK, and diskette drives before, do not worry; Section 2 explains what these terms mean. You will find this message more meaningful then.

1.2.4 System Prompt

After Concurrent CP/M-86 is loaded, you see this three-character message:

```
    0A>
```

This is the system prompt telling you Concurrent CP/M-86 is ready and waiting to read a command from your keyboard. It also tells you that drive A is your default drive. This means that until you tell Concurrent CP/M-86 to do otherwise, it first looks for your program and data files on the diskette in drive A. If the file is not found Concurrent CP/M-86 looks elsewhere, following the search rules described in Section 3.1.

1.3 Concurrent CP/M-86 Commands

Concurrent CP/M-86 performs certain tasks according to commands that you type at your keyboard. A Concurrent CP/M-86 command line is composed of a command keyword, an optional command tail, and a carriage return keystroke. The command keyword identifies a command (program) to be executed. The command tail can contain extra information for the command such as a filename or parameter. To end the command line, you must press the carriage return key, ↵.

As you type characters at the keyboard, they appear on your screen and the cursor moves to the right. If you make a typing error, push the backspace key, ←, to move the cursor to the left and correct the error.

You can type the keyword and command tail in any combination of upper-case and lower-case letters. Concurrent CP/M-86 treats all letters in the command line as upper case.

Generally, you type a command line directly after the system prompt. However, Concurrent CP/M-86 does allow spaces between the prompt and the command keyword.

A command keyword identifies commands. Concurrent CP/M-86 commands are actually the names of program files stored on your system diskette. They must be loaded into memory to perform their task.

When you request a command program, Concurrent CP/M-86 checks the command keyword. If you include a command tail, Concurrent CP/M-86 passes it to the utility without checking it because many utilities require unique command tails.

Let us look at how Concurrent CP/M-86 serves a request to run DIR. This command tells Concurrent CP/M-86 to display the names of diskette files on your screen. Type the DIR keyword after the system prompt, omit the command tail, and press carriage return:

```
    0A>DIR
```

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

```
DIR      CMD
FUNCTION CMD
STAT     CMD
```

Concurrent CP/M-86 recognizes only correctly spelled command keywords. If you make a typing error and press carriage return before correcting your mistake, Concurrent CP/M-86 echoes the command line with a question mark at the end. For example, if you accidentally mistype the DIR command, Concurrent CP/M-86 responds,

```
0A>DJR
DJR:    ?Can't find Command
```

telling you that it does not understand the command keyword.

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

```
0A>DIR PIP.CMD
```

Concurrent CP/M-86 performs this task by writing either the name of the file you specified or, if the file is not on the diskette, the message FILE NOT FOUND.

Be sure to type at least one space after DIR to separate the command keyword from the command tail. If you do not, Concurrent CP/M-86 responds with:

```
0A>DIRPIP.CMD
DIRPIP.CMD?
```

Some of the utility programs display messages that require a response. When you type your answer, you must press the carriage return key to send the response to the program.

1.4 Concurrent CP/M-86 Line-editing Control Characters

You can correct simple typing mistakes with the backspace, ←, key. However, Concurrent CP/M-86 provides the following control character commands to help you edit more efficiently. You can use these control characters to edit command lines or input lines to most programs if you want to change them before pressing carriage return. To type a control character, hold down the CTRL key and press the required letter key. Release both keys. The following table lists the Concurrent CP/M-86 control characters.

Table 1-1. Concurrent CP/M-86 Control Characters

Control Character	Meaning
CTRL-E	moves the cursor to the beginning of the following line without erasing your previous input.
CTRL-H	moves the cursor left one character position and deletes the character. The same as the backspace (←) key.
CTRL-I	moves the cursor to the next tab stop, where tab stops are automatically placed at each eighth column. Same as the ↹ key.
CTRL-J	moves the cursor to the left of the current line and sends the command line to Concurrent CP/M-86. Same as a carriage return (↵) keystroke.
CTRL-M	moves the cursor to the left of the current line and sends the command line to Concurrent CP/M-86. Same as an carriage return (↵) keystroke.
CTRL-R	types 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 that you have typed so far, types a # at the current cursor position, and moves the cursor to the next command line.
CTRL-X	discards all the characters in the command line that you have typed so far and moves the cursor back to the beginning of the current line.

Some control characters have the same meaning. For example, the CTRL-J and CTRL-M keystrokes have the same effect as pressing the carriage return key. All three send the command line to Concurrent CP/M-86 for processing. Also, pressing CTRL-H has the same effect as pressing the backspace (←) key.

End of Section 1

Section 2
Files, Diskettes, Drives, and Devices

Concurrent CP/M-86's most important task is to access and maintain files on your diskettes. It can create, read, write, and erase program and data files. This section tells you what a file is, how to create, name, and access a file, and how files are stored on your diskettes. It also tells how to indicate to Concurrent CP/M-86 that you have changed diskettes or that you want to change your default drive.

2.1 What is a File?

A Concurrent CP/M-86 file is a collection of related information stored on a diskette. Every file must have a unique name because Concurrent CP/M-86 uses that name to find that file. A directory is also stored on each diskette. The directory contains a list of the file specifications stored on that diskette and the locations of each file on the diskette.

In general, there are two kinds of files: program files and data files. A program file is an executable file, a series of instructions the computer can follow 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. For example, your computer cannot execute names and addresses, but it can execute a program that prints names and addresses on mailing labels.

A data file can contain the source code for a program. Usually, a program source file must be processed by an Assembler or Compiler before it becomes an executable program file. In most cases, an executing program processes a data file. However, there are times when an executing program processes an executable program file. For example, the DDT-86 program can both edit and execute a command file.

2.2 How Are Files Created?

There are many ways to create a file. You can create a file by copying an existing file to a new location, perhaps renaming it in the process. With Concurrent CP/M-86, you can use the PIP utility program to copy and rename files. Another way to create a file is by using a text editor. The Concurrent CP/M-86 text editor ED creates files and assigns them the names you specify. Finally, some programs, such as ASM-86, create output files as they process input files.

2.3 How Are Files Named?

Concurrent CP/M-86 identifies every file by its unique file specification. A file specification can have four parts: a drive specification, a filename, a filetype, and a password. We recommend that you create file specifications made up of letters and numbers.

A file specification can be simply a one to eight character filename, such as:

```
MYFILE2
```

When you make up a filename, choose a name that tells you something about what the file contains. For example, if you have a list of customer names for your business, you could name the file,

```
CUSTOMER
```

so that the name is eight or fewer characters and also gives you some idea of what is in the file.

As you begin to use your IBM Personal Computer with Concurrent CP/M-86, you find that files fall naturally into families. To keep file families separated, Concurrent CP/M-86 allows you to add a family name, called a filetype, to the filename. For example, you could add the following filetype to the file that contains a list of customer names:

```
CUSTOMER.NAM
```

The executable program files that Concurrent CP/M-86 loads into memory from a diskette have different filenames, but are in the family of programs that run with Concurrent CP/M-86. The filetype .CMD identifies this family of executable programs.

A filetype can be up to three characters long. Try to use three letters that tell something about the file's family. When you type a file specification for Concurrent CP/M-86 to read, separate the filetype from the filename with a period. When Concurrent CP/M-86 lists file specifications for you to read in response to a DIR command, it separates the filename from the filetype with blanks so that you can compare filetypes quickly.

Concurrent CP/M-86 has already established several file families. Table 2-1 lists their filetypes with a short description of each family.

Table 2-1. Concurrent CP/M-86 Filetypes

Filetypes	Description
.CMD	8088 Machine Language Program
.LST	Printable output from ASM-86
.$$$	Temporary
.A86	ASM-86 Source Program
.H86	Assembled ASM-86 Program in hexadecimal format
.SUB	List of commands to be executed by SUBMIT

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

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

Note: the Syntax descriptions in Section 4 use the term filespec to indicate any valid combination of the elements included in the file specification: a drive specification, a primary filename, and a filetype. Valid combinations are:

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

The characters in the Table 2-2 have special meanings in Concurrent CP/M-86. Do not use these characters in file specifications. All other characters are allowed.

Table 2-2. Special Characters

Character	Meaning
< = , tab space carriage return	file specification delimiters
:	drive delimiter in file specification
.	filetype delimiter in file specification

Table 2-2. (continued)

Character	Meaning
;	password delimiter in file specification
* ?	wildcard characters in file specification
< > & !	reserved
[ ]	option list delimiters
( )	delimiters for multiple modifiers in option list
/ $	option delimiters
;	comment delimiter in column one

The less than, equal, comma, tab, space, and carriage return characters separate file references and other items in the command line.

The colon and period separate drive specifications and filetypes in file specifications.

A semicolon in a file reference delimits a password.

The asterisk and question mark characters, * and ?, are wildcard characters in ambiguous file specifications (see Section 2.4.1).

The less than and greater than characters, < and >, are reserved for future use.

Square brackets, [and], isolate an option or option list from its command keyword (global option), or from its file specification (local option).

Parentheses, (and), isolate a list of more than one modifier, inside the square brackets, for options that have modifiers (see the SDIR utility in Section 4.19).

The slash, /, and dollar sign, $, are reserved for the specification of options in the command line.

A semicolon at the beginning of a command line indicates that the line is a comment.

2.4 How Are Files Accessed?

When you type a file specification in a command tail, the utility looks for the file on the diskette in the drive named by the system prompt, called the default drive. For example, if you type the command,

```
0A>dir myprog.lst
```

Concurrent CP/M-86 looks in the directory of the diskette in drive A for MYPROG.LST. If not found there, DIR searches the system drive for the specified file (Section 3.1 describes the search pattern in detail).

You also need a way to tell Concurrent CP/M-86 to access the diskette in another drive, if your file happens to be elsewhere. For this reason, Concurrent CP/M-86 lets you precede a filename with a drive specification, which is the drive letter followed by a colon. For example, in response to the command,

```
0A>dir b:myfile.lib
```

Concurrent CP/M-86 looks for the file MYFILE.LIB in the directory of the diskette in drive B.

You can also precede an executable program filename with a disk specification, even if you are using the program filename as a command keyword. For example, if you type,

```
0A>b:sort.cmd
```

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

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

Concurrent CP/M-86 handles programs with the SYS attribute, such as PIP, DIR, and other .CMD files residing on your system diskette, somewhat differently. Section 3.1 explains how Concurrent CP/M-86 looks around to find these programs, even if you do not specify the drive they reside in.

2.4.1 Accessing More Than One File: Wildcard Characters

Certain Concurrent CP/M-86 utilities select and process several files when special wildcard characters are included in the filename or filetype. A file specification containing wildcards can refer to more than one file because it gives Concurrent CP/M-86 a pattern to match. Concurrent CP/M-86 searches the diskette 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 rules for using wildcards are listed below.

- A ? matches any character in a name, including a space character.
- A * must be the last, or only, character in the filename or filetype. Concurrent CP/M-86 internally replaces a * with ? characters to the end of the filename or filetype.
- When the filename to match is shorter than eight characters, Concurrent CP/M-86 treats the name as if it ends with spaces.
- When the filetype to match is shorter than three characters, Concurrent CP/M-86 treats the filetype as if it ends with spaces.

Suppose, for example, you have a diskette with six files named:

A.CMD, AA.CMD, AAA.CMD, B.CMD, A.A86, and B.A86

Several cases are listed below where a name with wildcards matches all, or a portion of, these files:

.	is treated as ????????.???
????????.???	matches all six names
*.CMD	is treated as ????????.CMD
????????.CMD	matches the first four names
?.CMD	matches A.CMD and B.CMD
?.*	is treated as ?.???
?.???	matches A.CMD, B.CMD, A.A86, and B.A86
A?.CMD	matches A.CMD and AA.CMD
A*.CMD	is treated as A???????.CMD
A???????.CMD	matches A.CMD, AA.CMD, and AAA.CMD

Remember that Concurrent CP/M-86 uses wildcard patterns only while searching a diskette directory. Therefore, wildcards are valid only in filenames and filetypes. You cannot use a wildcard in a drive specification.

2.4.2 Start-up Files

You can execute a command automatically upon loading the system if you provide a start-up file for a particular virtual console. Each virtual console accepts a one-line command contained in a start-up file automatically loaded and executed when the system begins operation.

This command string is any Concurrent CP/M-86 command such as DIR, STAT, or VCMODE. If you want to execute more than one command, you can specify the SUBMIT command with a filespec, and Concurrent CP/M-86 then reads and executes the contents of your specified SUBMIT file. This SUBMIT file can be as long and complex as you want. Once the single line or SUBMIT file has finished execution, you see the familiar system prompt reappear, and you can enter further instructions.

When you boot the system, Concurrent CP/M-86 looks for a start-up file on drive A with the name n.SUP, where n is the number of a virtual console: 0.SUP controls console 0, 1.SUP controls console 1, and so forth.

Create these start-up files with ED, the Concurrent CP/M-86 text editor (described in Section 5), or with another word processor. If you issue a SUBMIT command to direct execution from a SUBMIT file, follow the SUBMIT rules described in Section 4.23. You can not SUBMIT the .SUP file itself; you can only load this file automatically on start up.

Start-up files are useful. For example, you can automatically reset the background mode for an individual virtual console, redefine the function keys to support a particular word processor, set serial printer characteristics, or transfer command files from your system disk to the MDISK. Sophisticated users can create start-up files that load automatically and relieve less knowledgeable users from such programming tasks.

2.5 Passwords

Concurrent CP/M-86 lets you protect files with passwords. If more than one person uses your system, passwords enable all users to protect their files from accidental damage by other users. Passwords also let managers and systems personnel limit access to a particular file for security.

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

The PASSWORD option of the SET command assigns a password to any file (see Section 4.20). All executable programs, commands, and data files can have individual password protection. Furthermore, the commands ED, ERA, ERAQ, PIP, REN, and TYPE accommodate passwords with your specified filename. In these cases, a command line can require multiple passwords to execute properly. The first password permits access to the command program; the second password permits access to the file specified in the command tail. In the following examples of command lines with passwords, assume that all files have been assigned the password XYZ.

```
0A>TYPE;XYZ PAYROLL.LST;XYZ
0A>TYPE;XYZ B:CAT.ASM;XYZ
0A>REN;XYZ NEWNAME.TYP = OLDNAME.TYP;XYZ
0A>ED;XYZ DOCUMENT.LAW;XYZ
0A>ERA;XYZ C:*.*;XYZ
```

Some Concurrent CP/M-86 commands and most word processing, accounting packages, and other applications programs running under Concurrent CP/M-86 do not accept passwords in the command tail. If you want to access password-protected files without typing the password each time you access the file, set the default password before executing the application program. For example, if you first issue the following set default password command, you do not have to specify the password XYZ in the above examples (See the SET command in Section 4.20).

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

Concurrent CP/M-86 displays an error message when you violate password syntax usage. For example, the following example fails to supply a necessary password:

```
0A>TYPE CLOCK.A86
```

This example gives the XYZ password in the TYPE invocation, but lacks the XYZ password in the filespec:

```
0A>TYPE;XYZ READQUE.A86
```

This example shows a mistyped password in the TYPE invocation:

```
0A>TYPE;XYX INITIAL.A86;XYZ
```

All of these incorrect commands return the following error message:

```
Bdos Err On d: Password Error
Bdos Function: 15    File:  FILENAME.TYP
```

Passwords can contain any characters except those listed in Table 2-2, Special Characters, in Section 2.3. All passwords are converted to upper-case when entered in file specifications or in the standard Concurrent CP/M-86 utilities. Application programs using the password protection features of Concurrent CP/M-86, however, can distinguish between upper- and lower-case passwords.

2.6 How Are Files Organized And Protected?

Under Concurrent CP/M-86 you can organize your files into groups, protect your files from accidental change, and specify how your files are displayed in response to a DIR command. Concurrent CP/M-86 supports these features by assigning user numbers and attributes to files and recording them in the diskette directory.

2.6.1 User Numbers

You can use user numbers to separate your files into 16 file groups. All files are identified by a user number that ranges from 0 to 15. Concurrent CP/M-86 assigns a user number to a file when the file is created. Generally, the file is assigned the current user number. You can change the current user number with the USER command described in Section 4.27.

Most commands can access only those files that have the current user number. For example, if the current user number is 7, a DIR command displays only the files that were created under user number 7. The exception to this is the PIP command. PIP can copy a file with one user number and send the copy to another user number. See the discussion of PIP in Section 4.16.

2.6.2 File Attributes

File attributes control how a file can be accessed. There are two kinds of file attributes. The first attribute can be either DIR (Directory) or SYS (System). When you create a file, it is automatically marked with the DIR attribute. You can display the name of a file marked with the DIR attribute only if the file has the current user number. If you give a file the SYS attribute, it is not displayed in response to a DIR command; you must use SDIR (described in Section 4.19).

If you give a file with user number 0 the SYS attribute, you can read and execute that file from any user number. This feature gives you a convenient way to make your commonly-used programs available under any user number. However, note that a user 0 SYS file does not appear in response to a SDIR command unless 0 is the current user number.

The second file attribute can be set to either R/W (Read-Write) or R/O (Read-Only). If a file is marked R/O, any attempt to write data to that file produces a Read-Only error message. Therefore, you can use the R/O attribute to protect important files. A file with the R/W attribute can be read or written to at any time, unless there is a tab over the write protect notch on the diskette, or unless the drive containing the diskette is set to Read-Only. You can use the STAT and SET programs to assign attributes to a file.

2.7 Back Up Your Files!

Humans have faults, and so do computers. Human or computer errors sometimes destroy valuable programs or data files. By mistyping a command, for example, you could accidentally erase a program that you just created. A similar disaster could result from an electronic component failure.

Data processing professionals avoid losing programs and data by making copies of valuable files, called making back-up copies or just making back-ups. Always make a working copy of any new program you purchase and save the original. If the program is accidentally erased from the working copy, you can easily restore it from the original.

Professionals also make frequent copies of new programs or data files when they are developing them. The frequency of making copies varies with each programmer, but as a general rule, make a copy at the point where it takes ten to twenty times longer to reenter the information than it takes to make the copy.

You can make back-ups in two ways. You can back up files one at a time, or you can can make a complete copy of the entire diskette. The choice is usually made based on the number of files on the diskette that need to be backed up. Copying an entire diskette takes little more time than backing up three or four files.

2.7.1 Single- and Double-Sided Diskettes

If you want to copy an entire diskette with DSKMAINT, you must copy between two diskettes of the same type: between two single-sided diskettes or between two double-sided diskettes. Look at the package of diskettes or at the diskette label to see if a particular diskette is single- or double-sided.

To make a double-sided copy of a single-sided diskette, follow the steps below to initialize your double-sided diskette. Then use PIP to copy individual files between the two dissimilar diskettes. The initialization process copies the necessary system loader tracks onto your new diskette.

2.7.2 Initializing Diskettes

When you take a new diskette out of its box, it is not ready to use yet. You must initialize (prepare) the diskette for use with the Personal Computer. If you have an old diskette that you want to reuse, follow these instructions to remove all the old data on the diskette and reformat it as if it were new.

First, the diskette must be formatted so the Personal Computer can read and write data to it in an orderly fashion. The diskette must be verified to ensure that the formatting information is written correctly. Finally, the diskette receives the system information needed to make Concurrent CP/M-86 run. The DSKMAINT utility does all of this with one subcommand.

Before you call DSKMAINT, be sure your current virtual console is in Dynamic mode. Set it to Dynamic mode with VCMODE. When you give the command,

```
0A>dskmaint
```

the screen in Figure 2-1 appears. Once this screen appears, you can remove diskettes from the drives. Place your uninitialized diskette in the appropriate drive.

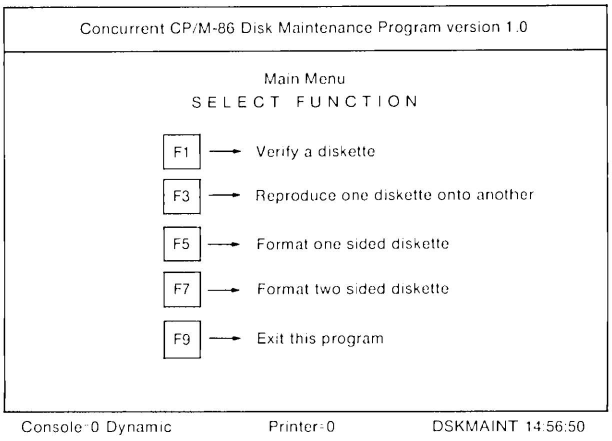
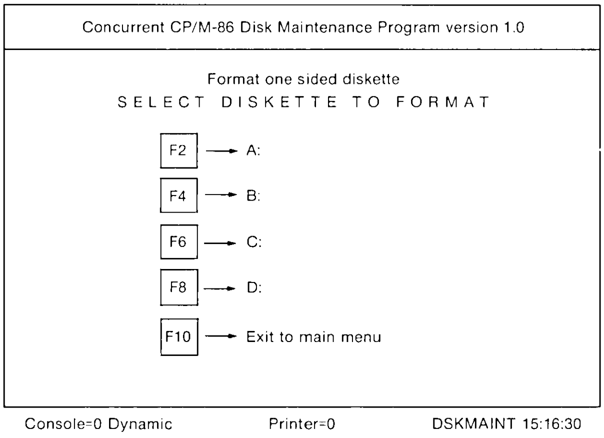

Concurrent CP/M-86 Disk Maintenance Program version 1.0
Main Menu
SELECT FUNCTION
F1 → Verify a diskette
F3 → Reproduce one diskette onto another
F5 → Format one sided diskette
F7 → Format two sided diskette
F9 → Exit this program
Console 0 Dynamic
Printer=0
DSKMAINT 14:56:50

Figure 2-1. DSKMAINT Main Menu

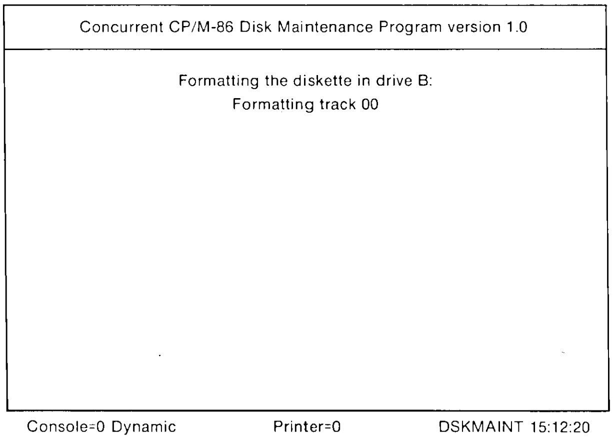
This is called the DSKMAINT main menu. Your task is formatting one or more new diskettes, so press key F5 if you have a single-sided diskette, or F7 if you have a double-sided diskette. Pressing F5 shows the screen in Figure 2-2. If you pressed F7, the only difference is in the title, which reads "Formats two sided diskette".



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

              Format one sided diskette
          S E L E C T  D I S K E T T E  T O  F O R M A T

          [F2]  ---> A:
          [F4]  ---> B:
          [F6]  ---> C:
          [F8]  ---> D:
          [F10] ---> Exit to main menu

Console=0 Dynamic        Printer=0        DSKMAINT 15:16:30
```

Figure 2-2. Format One Sided Diskette

Press the key that specifies the drive containing your disk; F2 for drive A, F4 for drive B, and so forth. This screen disappears and the screen in Figure 2-3 appears:



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

          Formatting the diskette in drive B:
                Formatting track 00

Console=0 Dynamic        Printer=0        DSKMAINT 15:12:20
```

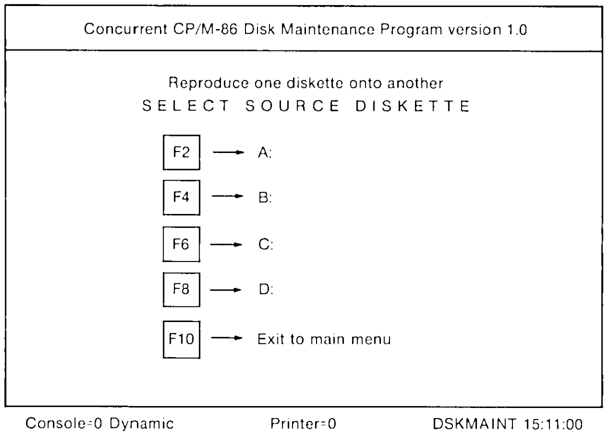
Figure 2-3. Formatting the Diskette

When DSKMAINT has finished formatting your diskette track by track, it automatically writes the Concurrent CP/M-86 system loader information to it, then verifies the diskette contents track by track, counting down the tracks as it works. The "Formatting track n" message changes to "Verifying track n". When the verification is through, the screen returns to the main menu.

Now that you have one or more new initialized diskettes, you are ready to duplicate your diskettes.

2.7.3 Copying Diskettes

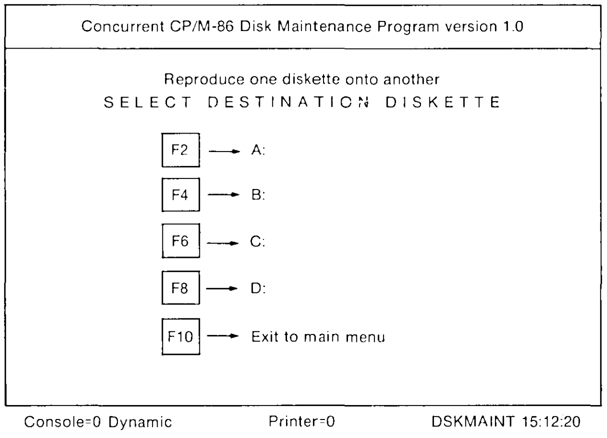
Once you have called DSKMAINT as described above and replaced old diskettes with the diskettes you want to copy across, press F3 to copy diskettes. The screen in Figure 2-4 appears:



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

Reproduce one diskette onto another
SELECT SOURCE DISKETTE

F2  --> A:
F4  --> B:
F6  --> C:
F8  --> D:
F10 --> Exit to main menu

Console=0 Dynamic        Printer=0        DSKMAINT 15:11:00
```

Figure 2-4. Reproduce Diskette: Select Source Diskette

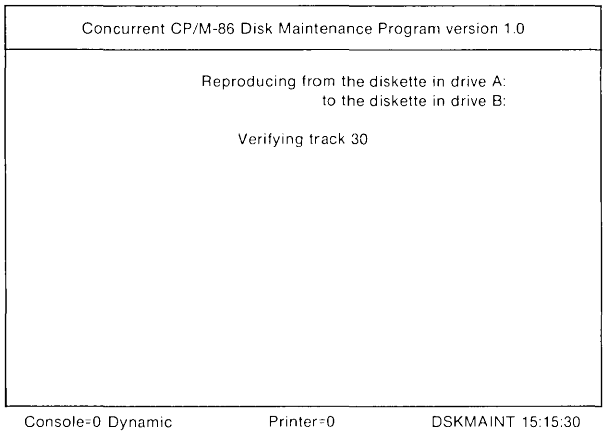
This screen asks you to specify which drive contains the source diskette with the original files to copy. Press the appropriate key; F2 for drive A, F4 for drive B, and so forth. This screen is replaced by the screen in Figure 2-5, which asks you to specify the drive containing the destination diskette. The destination diskette receives the copied files.



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

Reproduce one diskette onto another
SELECT DESTINATION DISKETTE

F2  → A:
F4  → B:
F6  → C:
F8  → D:
F10 → Exit to main menu

Console=0 Dynamic        Printer=0        DSKMAINT 15:12:20
```

Figure 2-5. Reproduce Diskette: Select Destination Diskette

As the copying operation takes place DSKMAINT informs you of its progress. The screen in Figure 2-6 tells you which disk is the source and which is the destination, noting the tracks as they are written, verified, and then reread to confirm proper operation.



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

Reproducing from the diskette in drive A:
                to the diskette in drive B:

Verifying track 30

Console=0 Dynamic        Printer=0        DSKMAINT 15:15:30
```

Figure 2-6. Reproducing the Diskette

After DSKMAINT finishes copying, the main menu returns to the screen.

2.7.4 DSKMAINT Error Messages

Before you call DSKMAINT, be sure your current virtual console is in Dynamic mode. Set it to Dynamic mode with VCMODE.

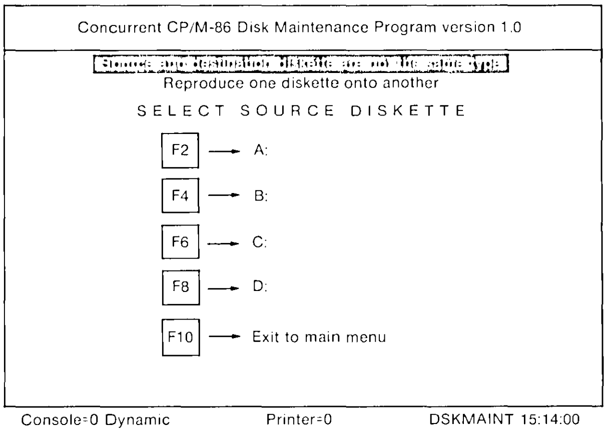
You might see an error message while performing a DSKMAINT operation; these messages appear in the status line at the bottom of your screen. This message is:

```
Disk Error  Accept/Retry/Ignore/Details?
```

Type A for Accept. DSKMAINT returns to the main menu and shows the BAD DISK TRY ANOTHER ONE error message printed in reverse video (black letters on a green strip) in the location shown in Figure 2-7. Remove the appropriate diskette, replace it with another, and try the operation again.

If you have double- and single-sided diskettes in your drives and try to copy across them, you obtain the following error message shown in the same location and format as the BAD DISK message above:

```
Source and destination diskette are not the same type
```



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

Source and destination diskette are not the same type
Reproduce one diskette onto another
SELECT SOURCE DISKETTE

F2  --> A:
F4  --> B:
F6  --> C:
F8  --> D:
F10 --> Exit to main menu

Console=0 Dynamic        Printer=0        DSKMAINT 15:14:00
```

Figure 2-7. Source and Destination Diskettes not Same Type

2.8 How Are Files Stored On Diskettes?

Concurrent CP/M-86 stores files on a single-sided or double-sided diskette in the same manner. Concurrent CP/M-86 records the filename, filetype, user number, and attributes of each file in a special area of the diskette called the directory. In the directory, Concurrent CP/M-86 also records which diskette sectors belong to which file, and which diskette sectors are still free. The directory is large enough to store this data for up to 64 files, whether the diskette is single-sided or double-sided.

Concurrent CP/M-86 sets aside directory and storage space for a file only as the file grows. When you erase a file, Concurrent CP/M-86 reclaims this allocated space in two ways: it makes the file directory space available to catalog a different file, and it frees the file storage space for later use.

This dynamic allocation feature makes Concurrent CP/M-86 powerful. You do not have to tell Concurrent CP/M-86 how big your file will become, because Concurrent CP/M-86 automatically allocates more storage for a file as it is needed and releases the storage for reallocation when the file is erased.

2.9 Changing Diskettes

Concurrent CP/M-86 cannot, of course, do anything to a file unless the diskette that holds the file is inserted into a drive and the drive door is closed. When a diskette is in a drive, it is on-line, and Concurrent CP/M-86 can read its directory.

At some time, you have to take a diskette out of a drive and insert another that contains different files. You must tell Concurrent CP/M-86 that you changed a diskette by giving the DSKRESET command directly after the system prompt. In response, Concurrent CP/M-86 logs in the new diskette.

If you forget to issue the DSKRESET command after you change a diskette, Concurrent CP/M-86 automatically protects the new diskette by setting it to Read-Only (R/O). You can run a text editor or copying program and try to write to the new diskette, but when you do, Concurrent CP/M-86 notices that the original diskette is no longer in the drive and writes the message,

```
Bdos Err on d: R/O
Bdos Function : nn    File:   filespec.typ
```

where d: is the drive specification of the new diskette. If you get this message, type DSKRESET after the system prompt returns.

There are times when it is appropriate to load a command program into memory and then change a diskette while the program in memory has paused for the change. At these times, the new diskette that you insert in the drive must be of the same type as the diskette you have removed from the drive. If it is not of the same type, Concurrent CP/M-86 cannot read it properly.

To sum up, there are two things to note when changing a diskette after loading a program into memory:

- You must replace the diskette you removed with a diskette of the same type, either single-sided or double-sided.
- Concurrent CP/M-86 can read from the changed diskette but cannot write to a diskette unless it has been logged in at system start-up or with a DSKRESET following the system prompt.

Whenever you have one or more virtual consoles operating in the Buffered mode, Concurrent CP/M-86 opens and maintains one or more buffer files dedicated to these consoles. These files retain data output to the screen by the program running on the buffered console. When you switch virtual consoles and bring a background console into the foreground, all of this data can be written to the screen.

Concurrent CP/M-86 puts these buffer files on your system drive, which is your highest-lettered drive: on a two-drive system, in drive B; on a three-drive system, drive C; on a four-drive system, drive D.

However, if you installed memory into the memory locations reserved for MDISK (described in detail in Section 2.11.1), Concurrent CP/M-86 uses this memory for buffered file storage. You can then use your system diskette like any other.

Do not remove your system diskette from its drive while any virtual consoles are in Buffered mode with open files. If you do remove this diskette, programs on a background Buffered-mode console cease execution until switched to the foreground.

If you try to replace your system diskette with another diskette, Concurrent CP/M-86 checks the new diskette's directory and decides it is new. In this case, no buffering occurs until the drive is set to Read/Write with the DSKRESET command.

To avoid this, use the DSKRESET command (explained in Section 4.9). If you want to remove your system diskette, give the DSKRESET command first. If any buffered console files are open, Concurrent CP/M-86 returns an error message similar to this example,

```
Disk reset denied, Drive B: Console 2  Program VOUT2
```

and you must wait until these files close.

2.10 Changing The Default Drive

At any given time during Concurrent CP/M-86 operation, there is one drive called the default drive associated with each virtual console. Unless you put a drive specification in your command line, Concurrent CP/M-86 and the utilities look in the directory of the diskette in the default drive for all of your program and data files. You can tell the default drive by looking at the Concurrent CP/M-86 system prompt. For example, the message:

```
0A>
```

tells you that the A drive is the default drive. When you give commands to Concurrent CP/M-86, remember which diskette is the default drive. This way, you know which files an application program can access if you do not add a drive specification.

Drive A is the default drive when you start Concurrent CP/M-86. If you want to change the default drive, type the drive specification of the desired default drive next to the system prompt and press the carriage return key.

```
0A>B:
```

The command in the preceding example changes the default drive to B. Unless you change the default drive again, all system prompt messages appear as:

```
0B>
```

The system prompt now indicates that Concurrent CP/M-86 and its utilities check in the directory of the diskette in drive B for any file that does not have a drive specification included in the file specification.

2.11 More Concurrent CP/M-86 Drive Features, MDISK Virtual Drive

Under Concurrent CP/M-86, drives can be marked R/O, just as files can be given the R/O attribute. The default state of a drive is R/W, but Concurrent CP/M-86 marks a drive R/O whenever you change the diskette in the drive. To return the drive to R/W, you must type DSKRESET following the system prompt. You can give a drive the R/O attribute by using the SET command, (described in Section 4.20).

MDISK is a virtual diskette drive. This means that MDISK acts like a diskette drive as far as Concurrent CP/M-86 is concerned; you can PIP files to and from MDISK look at the MDISK directory with DIR, and perform all the commands you normally use with your diskette drives except for copying diskettes.

MDISK cannot contain diskettes. MDISK is an area in Personal Computer memory that emulates a very fast diskette drive. You determine how much storage space MDISK has by placing memory in its area in 64K byte increments. You do not have to use the MDISK area, but if you want to, you must have at least 64K bytes of RAM and no more than 192K bytes residing there.

To implement MDISK, you need to place a Random Access Memory (RAM) card in your system. This memory card should carry switches that determine where its memory is located in the Personal Computer

Random Access Memory area. Refer to the instructions accompanying this card and set these switches so the memory begins at the address C000:0.

We recommend that you do use MDISK, and that you use at least 128K bytes of RAM. This allows storage of an entire diskette-worth of information in MDISK. If you have a two-drive Personal Computer and you use MDISK, you can PIP your command files to MDISK, freeing drive B. In this case, Concurrent CP/M-86 writes buffered console output into files contained in MDISK, and does not maintain these files on your system diskette in drive B. You can then remove your system diskette without worrying about losing buffered console data.

2.12 Other Concurrent CP/M-86 Devices

Concurrent CP/M-86 manages all the peripheral devices attached to your IBM Personal Computer. Peripheral devices are actual physical pieces of hardware such as diskette drives, input devices such as keyboards, and output devices such as printers, serial I/O devices, and CRT terminals.

A logical device is a name that Concurrent CP/M-86 uses to keep track of input and output. The following list shows Concurrent CP/M-86 logical device names and indicates whether the device is an input or output device.

CON:	Console input and output
LST:	List output

Note: Concurrent CP/M-86 does not support auxiliary input and output, AXI: and AXO, respectively.

End of Section 2

Section 3
Concurrent CP/M-86 Command Concepts

As we discussed in Section 1, a Concurrent CP/M-86 command line consists of a command keyword, an optional command tail, and carriage return keystroke. This section describes the kinds of programs the command keyword identifies, and tells how Concurrent CP/M-86 searches for command files on a diskette. It also introduces the control characters and function keys that direct Concurrent CP/M-86 to perform various tasks.

3.1 How Concurrent CP/M-86 Searches for Commands

When you type a command line on the IBM Personal Computer, Concurrent CP/M-86 looks on the default diskette or specified diskette for a program file. It looks for a filename equal to the keyword and a filetype of .CMD. For example, suppose you type the command line:

```
0A>ED MYPROG.BAS
```

Concurrent CP/M-86 goes through these steps to execute the command:

1) Concurrent CP/M-86 searches for the utility program file ED.CMD in the directory of the default drive. If it does not find the file under the current user number, it looks under user number 0 for ED.CMD with the SYS attribute, and on the system drive.

2) When Concurrent CP/M-86 locates ED.CMD, it copies the program into memory and passes control to ED.

3) ED runs in memory until you enter a command to exit ED.

4) Concurrent CP/M-86 types the system prompt and waits for you to type another command line.

If Concurrent CP/M-86 cannot find the requested .CMD program, it reports an error in this way:

```
XXX: ?Can't Find Command
```

This tells you that no corresponding .CMD file appears under the current user number, or with the SYS attribute under user 0 on the default, specified, or system drive.

For example, suppose your default diskette contains only standard Concurrent CP/M-86 utilities and you type the following command line:

```
0A>EDIT MYPROG.A86
```

These are the steps that Concurrent CP/M-86 goes through to report the error:

1) Concurrent CP/M-86 first searches the default drive for EDIT.CMD, under the current user number. If Concurrent CP/M-86 cannot find EDIT.CMD, it looks in user 0 for EDIT.CMD with the SYS attribute enabled.

2) The system then searches the system diskette directories under the current user number. Finally, if EDIT.CMD still cannot be found, Concurrent CP/M-86 searches the system drive under user 0.

3) If the file cannot be found, Concurrent CP/M-86 writes the message,

   ```
   EDIT:  ?Can't Find Command
   ```

   to tell you that the command cannot be executed.

4) Concurrent CP/M-86 displays the system prompt and waits for you to type another command line.

3.2 Control Character Commands

You can direct Concurrent CP/M-86 to perform certain functions just by striking special keys. With the control character commands, you can tell Concurrent CP/M-86 to start and stop screen scrolling or echo the screen display at the printer. The following table summarizes Control Character Commands.

Table 3-1. Control Character Commands

Control Character	Meaning
CTRL-C	prompts you for termination of the currently operating program.
CTRL-F	flushes all screen output returned when a buffered console is switched-in to the physical console. Once deleted, this data cannot be recovered.
CTRL-P	tells Concurrent CP/M-86 to send screen output to the printer and to the screen. If the printer is already echoing the screen, CTRL-P tells Concurrent CP/M-86 to stop sending screen output to the printer.

Table 3-1. (continued)

Control Character	Meaning
CTRL-Q	enables screen scrolling after you halt it with CTRL-S. Use these two commands to halt screen displays that pass by too rapidly.
CTRL-S	stops screen scrolling. If the display on your screen rolls by too quickly for you to read, press CTRL-S. Press CTRL-Q to continue the display.
CTRL-Break	works like CTRL-C. Pressing CTRL-Break prompts you for termination of the currently operating program.
PrtSc	works like CTRL-P. Directs screen output to the printer and halts screen output to the printer. CTRL-PrtSc also works in the same way.

3.3 Programmable Function Keys

There are twenty Programmable Function Keys (PFKs) on the Personal Computer keyboard. You can define each of these keys from each individual virtual console; to program function keys for virtual console 1 you must first make console 1 the current (switched-in) console by pressing CTRL and 1 at the same time. Once you specify a programmed function, these keys keep their definitions until you reload Concurrent CP/M-86.

You can program the keys with any valid Concurrent CP/M-86 command string up to nineteen characters long. Once programmed, pressing the key generates the entire command string as if you typed it after the system prompt.

These programmable keys are the ten keys labeled F1 through F10 on the left side of the keyboard, and ten keys in the number pad at the right side of the keyboard. The number pad keys are those labeled Home, ↑ , PgUp, ← , → , End, ↓ , PgDn, Ins, and Del.

The number pad keys have two functions: if you turn on the Num Lock key by pressing it (you see "Num" on the status line if Num Lock is on), pressing any of these numbered keys generates the number and pressing the shift key while pressing the same key returns the programmed function. When Num Lock is off, these keys return the programmed functions when pressed and the appropriate numbers when you press shift key and the appropriate key.

FUNCTION is the command that lets you specify functions for the programmable function keys. You can change their programming or simply view the current commands generated by the PFKs.

FUNCTION operates in 2 modes. Specify the mode by the command tail you enter with FUNCTION. If you supply no command tail, FUNCTION displays an interactive menu for you to select from. If you supply a filespec in the command tail, FUNCTION takes its input from the specified file.

Concurrent CP/M-86 comes up with commands programmed into the ten F1-F10 function keys at the left of the keyboard--that is, Concurrent CP/M-86 is shipped with a start up file that specifies commands for these keys when the operating system is loaded. These commands are:

```
F1:  SDIR\0D
F2:  SDIR
F3:  STAT *.*\0D
F4:  STAT
F5:  VCMODE D\0D
F6:  VCMODE B\0D
F7:  DSKMAINT\0D
F8:  FUNCTION
F9:  HELP\0D
F10: DSKRESET\0D
```

The commands terminated with \0D above indicate that these functions include carriage returns. When you press the key, Concurrent CP/M-86 acts as if you had given the command with a carriage return. The commands without \0D can take further specifications or parameters you supply. Terminate them with a carriage return and Concurrent CP/M-86 carries out the command.

As noted earlier, these commands are programmed by a start-up file (N.SUP) loaded and executed when Concurrent CP/M-86 starts up. However, you can change this start-up file, replace it, or eliminate it altogether and still retain access to this set of programmed commands; pressing the shift key and any of the 10 function keys produces the command.

3.4 Concurrent CP/M-86 Command Summary

This section provides a table of Concurrent CP/M-86 commands listed alphabetically.

Table 3-2. Command Summary

Command	Function
ABORT	interrupts program execution on the specified virtual console.
ASM86	translates 8086 assembly language programs into hex format.
CONFIG	programs serial port operation.
DDT86	helps you check out your assembly language programs and interactively correct bugs and programming errors.
DIR	displays a list of file specifications from a diskette directory. DIR by itself does not show files with the [SYS] attribute set. Enter the DIR [S] or SDIR commands to see these files.
DSKMAINT	lets you copy, verify, and format your diskettes.
DSKRESET	logs in diskettes when you first place them in a diskette drive.
ED	lets you create and alter text or data files a line at a time.
ERA	erases one or more file specifications from a diskette directory and releases the storage occupied by the file.
ERAQ	erases one or more file specifications from a diskette directory, as ERA does, but ERAQ asks you to confirm the command.
FUNCTION	programs the Function Keys and the Numeric Keypad on your keyboard.
GENCMD	uses ASM-86 output to produce an executable command file.
HELP	displays information on how to use Concurrent CP/M-86 commands.
PIP	combines and copies files.
PRINTER	shows the currently assigned printer, or assigns one.
REN	lets you rename a file.

Table 3-2. (continued)

Command	Function
SDIR	displays a diskette directory list of files and their attributes.
SET	lets you specify and alter certain file attributes.
SHOW	displays information about various system resources.
STAT	lets you examine and alter file status.
SUBMIT	sends a file of commands to Concurrent CP/M-86 for execution.
SYSDISK	sets the specified drive to system drive.
TOD	sets the date and time displayed in the status line at the bottom of your display.
TYPE	writes the contents of a character file at your screen.
USER	changes one user number to another.
VCMODE	switches virtual consoles between Buffered and Dynamic modes.

End of Section 3

Section 4
Command Summary

This section explains how to show the parts of a file specification in a command line. It also describes the notation used to indicate optional parts of a command line and other syntax notation. The remainder of the section is a reference for all standard Concurrent CP/M-86 commands. Each command is listed, followed by a short explanation of its operation with examples.

In the descriptions of some commands you see references to attached processes and detached processes. These concepts refer to the way Concurrent CP/M-86 keeps track of which virtual consoles are performing individual jobs. You do not need to know about attaching or detaching processes to do anything referred to in this but if you are interested in understanding these concepts, refer to the Concurrent CP/M-86 Operating System Programmer's Guide for further details.

The Concurrent CP/M-86 editor, ED, is described in detail in Section 5. The Concurrent CP/M-86 Operating System Programmer's Guide describes the commands and usage of the ASM-86 Assembler and the DDT-86 assembly-language debugging tool.

4.1 Command Specifications

You can see that there are several parts in a file specification that we must distinguish. To avoid confusion, we give each part a formal name that is used when we discuss command lines.

The four parts of a file specification are:

drive specifier	the optional diskette drive; a letter A, B, C, D, or M that contains the file or group of files to which you are referring. If a drive specification is included in your command line, it must be followed by a colon.
filename	the one- to eight-character first name of a file or group of files.
filetype	the optional one- to three-character family name of a file or group of files. The filetype must be separated from the filename by a period.
password	the optional one- to eight-character word that must be supplied to gain access to the specified file. Separate a password from the filename or filetype with a semicolon. Passwords are defined with the SET command.

The following indicates the general form of a file specification:

```
d:filename.typ
```

In the preceding form, d: represents the optional drive specification, filename represents the one- to eight-character filename, and .typ represents the optional one to three character filetype. The following are valid combinations of the elements of a Concurrent CP/M-86 file specification:

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

If you do not include a drive specification, Concurrent CP/M-86 automatically supplies the default drive. If you omit the period and the filetype, Concurrent CP/M-86 again automatically includes a filetype of three blanks.

This general form is called a file specification. A file specification names a particular file or group of files in the directory of the on-line diskette given by the drive specifier. For example,

```
B:MYFILE.A86
```

is a file specification that indicates drive B:, filename MYFILE, and filetype A86. File specification is abbreviated as simply

```
filespec
```

in the command syntax statements.

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

```
B:MY*.A??
```

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

You now understand command keywords, command tails, control characters, default drives, on-line drives, and wildcards. You also see how we use the formal names filespec, drive specification, filename, and filetype. These concepts give you the background you need to compose complete command lines on your own. We devote the next section to introducing the various command line forms.

4.2 How Commands Are Described

Section 4 lists the commands alphabetically. Each command description is given in a specific form. The description begins with the command keyword in upper-case. When appropriate, an English phrase that describes the command follows the keyword, in parentheses.

The Syntax section gives you one or more general forms to follow when you compose the command line.

The text describes the operation of each command, noting special cases, suggesting programming techniques, and describing special conventions and considerations.

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 syntax using these rules:

- Words in capital letters must be typed by you and spelled as shown, but you can use any combination of upper- or lower-case letters.
- You can substitute any number for n.
- The symbolic notation d:, filename, .typ and filespec have the general meanings described in the previous section.
- You must include one or more space characters where a space is shown, unless otherwise specified. For example, the PIP options need not be separated by spaces.
- Items enclosed in curly braces { } are optional. You can enter a command without the optional items. The optional items add effects to your command line.
- An ellipsis (...) tells you that the previous item can be repeated any number of times.
- When you can enter one or more alternative items in the syntax line, a vertical bar | separates the alternatives. Think of this vertical bar as the or bar.
- All other punctuation, such as square brackets [ ], must be included in the command line.

Let us look at some examples of syntax notation. The Concurrent CP/M-86 utility command SET specifies password protection and time stamping of files in the Concurrent CP/M-86 system. It also sets file and drive attributes, such as the Read-Only (R/O), Read-Write (R/W), System (SYS), Directory (DIR), and user-definable attributes.

The syntax section of the SET command shows how the command line syntax notation is used:

<u>Syntax:</u>

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

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

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

Using this syntax, we can construct several valid command lines:

```
SET B:[RW]
SET D:[RO]

SET B:INVOICE.LST [RW]
SET D:RESPOND.TXT [RO]
```

These command lines show that the Read-Write and Read-Only attributes can be set for entire drives or individual files.

```
SET PACK.A86  [SYS]
SET B:SORT    [DIR]
```

These command lines show that the System and Directory attributes require any valid filespecs for operation; PACK.A86 and B:SORT are valid filespecs.

```
SET [password=amber]
SET D:[password=amber, protection=write]
```

These command lines illustrate that SET options, such as PASSWORD and PROTECTION, can apply to entire drives. In the first example, the default drive is assigned the password AMBER, while in the second example, drive D is assigned the same password. The PROTECTION attribute specifies that you must give the correct password before Concurrent CP/M-86 allows you to WRITE to drive D.

```
SET DATABASE.LST [CREATE=ON]
SET PNTOFSAL.DAT [TIME]
```

These command lines demonstrate options applied to filespecs. The first example turns on CREATE time stamps for the DATABASE.LST file. The second example provides date/time stamps for the file PNTOFSAL.DAT.

The Concurrent CP/M-86 command PIP (Peripheral Interchange Program) is the file copy program. PIP can copy information from your screen to the diskette or printer. PIP can combine two or more files into one longer file. PIP can also rename files after copying them. Let us look at one of the formats of the PIP command line for another example of how to use command line notation.

Syntax:

```
PIP dest-filespec=source-filespec{,filespec...}
```

For this example, dest-filespec is further defined as a destination file specification or peripheral device (printer, for example) that receives data. Similarly, source-filespec is a file specification or peripheral device (keyboard, for example) that transmits data. PIP accepts wildcards in the filename and filetype. (See Section 4.16 for other capabilities of PIP.) Many valid command lines come from this syntax. Some are shown below.

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

4.3 The ABORT Command

Syntax:

```
ABORT programname
ABORT programname n
```

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

```
Abort Failed
```

The CTRL-C character also aborts a running program, but only if it is still attached to the console. When you press CTRL-C, the system temporarily halts the program and responds with the program name message:

```
XXX: Abort (Y/N)?
```

If you enter a Y, the program is immediately aborted. If you enter any other character, the program continues. Use the abort command with program running on another virtual console. Use CTRL-C with programs running on your current virtual console. If you abort more than one program on more than one virtual console, the ABORT commands are stored in the order given and the programs are interrupted one by one.

Example:

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

```
0A>TYPE DOCUMENT.TXT <cr>

Dear Sir:

The company is pleased to inform you tha

5B>ABORT TYPE 1 <cr>
5B>
```

In the preceding example, assume that the TYPE command was issued from virtual console 1. The TYPE command is aborted from virtual console 3.

4.4 The ASM86 (Assembler) Command

Syntax:

```
        ASM86  filespec { $parameter-list }
```

ASM-86 utility converts 8088 and 8086 assembly language source statements into machine code form.

The operation of the ASM-86 Assembler is described in detail in the Concurrent CP/M-86 Operating System Programmer's Guide. ASM-86 signs on as CP/M-86 ASM86.

The filespec names the character file that contains an 8086 assembly language program to translate. If you omit the filetype, a filetype of A86 is assumed. The Assembler uses the drive specification portion of the filespec as the destination drive for output files unless you include a parameter in the command tail to override this default.

The three output files produced by the Assembler are given the filetypes listed below:

- LST contains the annotated source listing.
- H86 contains the 8086 machine code in hex format.
- SYM contains all programmer-defined symbols with their program relative addresses.

The Assembler assigns the same filename as the source filename to the LST, H86, and SYM files.

You control the assembly process by including optional parameters in the parameter list. Each parameter is a single parameter letter followed by a single letter device name. The parameters can be separated by blanks, but each parameter letter must be followed immediately by the device name.

The parameter letters are A, H, P, S, and F. The device names are the letters A, B, C, D, and M, corresponding to the five drive letters. The letters X, Y, and Z have special meaning when used as device names:

- X is the Screen.
- Y is the Printer.
- Z is zero output.

Use the A parameter letter to override the default drive specification to obtain the source file. The valid parameters are AA, AB, AC, AD and AM.

Use the H parameter letter to override the default drive specification to receive the H86 file. Valid parameters are HA through HD, and HX, HY, and HZ.

Use the P parameter letter to override the default drive specification to receive the LST file. Valid parameters are PA through PD, and PX, PY, and PZ.

Use the S parameter letter to override the default drive specification to receive the SYM file. Valid parameters are SA through SD, and SX, SY, and SZ.

Use the F parameter letter to select the format of the hex output file. Valid parameters are FI and FD. The FI parameter selects Intel format hex file output. The FD parameter selects Digital Research format hex file output. FD is assumed if neither FI nor FD appear as a parameter. Use FI when the program is going to be combined with a program generated by an Intel Compiler or Assembler.

When conflicting parameters appear on the command line, the rightmost parameter prevails.

Passwords

ASM-86 does not support passwords. If your A86 files require passwords, be sure they all have the same one, and set the default password to this password.

Examples:

```
0A>ASM86 X
```

The ASM86.CMD file must be on drive A or the system drive. The source file X.A86 is read from drive A, and X.LST, X.H86, and X.SYM are written to drive A.

```
0B>ASM86 X.A86  $PX
```

The ASM86.CMD file must be on drive B or the system drive. The source file X.A86 is read from drive B. The listing is written to the screen, and the X.H86 and X.SYM files are placed on drive B.

```
0A>ASM86 B:MYPROG $PY HC
```

The source file MYPROG.A86 is read from drive B, the listing is sent to the printer, the file MYPROG.H86 is written to drive C, and file MYPROG.SYM is placed on drive B.

```
0A>B:ASM86 X $SZ
```

The ASM86.CMD file must be on drive B. The X.A86 file is read from drive A. The X.LST and X.H86 files are written to drive A. No X.SYM file is generated.

4.5 The CONFIG Command

<u>Syntax:</u>

```
        CONFIG
```

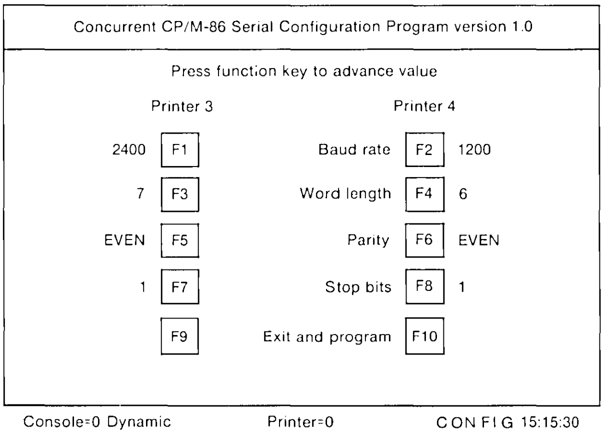
Concurrent CP/M-86 supports five printers: three parallel I/O printers designated printers 0, 1, and 2, and two serial I/O printers designated printers 3 and 4. The CONFIG command lets you specify the parameters of the serial printers in two different ways--interactively, and in a command line.

4.5.1 Interactive Mode

If you have two serial printers, when you enter

```
        0A>config
```

the screen in Figure 4-1 appears. If there is only one serial printer connected to your Personal Computer, only the Printer 3 column would appear. This screen shows how the function keys specify four sets of parameters for printers 3 and 4, odd-numbered keys for printer 3, and even-numbered keys for printer 4.



```
        Concurrent CP/M-86 Serial Configuration Program version 1.0

                    Press function key to advance value

              Printer 3                          Printer 4

        2400  [F1]               Baud rate  [F2]  1200

           7  [F3]             Word length  [F4]  6

        EVEN  [F5]                  Parity  [F6]  EVEN

           1  [F7]               Stop bits  [F8]  1

              [F9]        Exit and program  [F10]

  Console=0 Dynamic          Printer=0            CON FI G 15:15:30
```

Figure 4-1. Configuration Program

The values shown when you invoke CONFIG are the current values; you can change them by pressing the appropriate keys, each stroke increments the appropriate value. The F9 and F10 keys return you to command level.

The baud rate signifies the speed at which the computer sends data to its printer. Refer to your printer's technical manual for the correct baud rate to program.

The word length indicates the number of data bits in a single transmitted character. This value is almost always 7 or 8. Again, consult the printer's documentation to determine the correct value.

Parity monitors characters to see if they are transmitted and received correctly. Often both the printer and computer can enable or disable parity checking. Refer to the printer manual for specific details relevant to your printer.

Stop bits indicate the end of a transmitted character. See your printer's manual for the proper value.

4.5.2 Command Line Mode

If you want to define all serial port parameters in a single command line, CONFIG accepts such lines following this syntax:

```
CONFIG PORT# BAUDRATE WORDLENGTH PARITY STOPBITS
```

In this format, PORT# is either P3, indicating printer 3, or P4, indicating printer 4.

BAUDRATE is 110, 150, 300, 600, 1200, 2400, 4800, or 9600, indicating the transmission speed in baud (bits-per-second).

WORDLENGTH is 5, 6, 7, or 8, specifying the number of data bits for a transmitted character.

PARITY is either ODD, EVEN or NONE specifying the type of parity, if any, to be used.

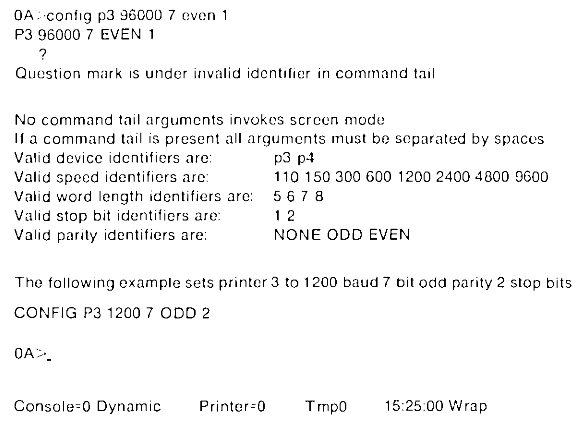
STOPBITS is either 1 or 2, adding that number of stop bits to the end of a transmitted character.

A typical CONFIG command line looks like this:

```
0A>config P3 9600 7 EVEN 1
```

This command line sets serial printer P3 to 9600 baud, with seven data bits, even parity and 1 stop bit.

If you make a mistake in a CONFIG command line, the screen in Figure 4-2 appears. The command line is returned with a question mark under the error, followed by a list of valid parameters.



```
0A>config p3 96000 7 even 1
P3 96000 7 EVEN 1
   ?
Question mark is under invalid identifier in command tail

No command tail arguments invokes screen mode
If a command tail is present all arguments must be separated by spaces
Valid device identifiers are:        p3 p4
Valid speed identifiers are:         110 150 300 600 1200 2400 4800 9600
Valid word length identifiers are:   5 6 7 8
Valid stop bit identifiers are:      1 2
Valid parity identifiers are:        NONE ODD EVEN

The following example sets printer 3 to 1200 baud 7 bit odd parity 2 stop bits

CONFIG P3 1200 7 ODD 2

0A>_

Console=0 Dynamic     Printer=0     Tmp0     15:25:00 Wrap
```

Figure 4-2. Error in CONFIG Command Line

If you have tried to configure a serial port on a system without such ports, CONFIG returns this error message:

```
No Serial communication options installed
```

and control returns to system command level.

4.5.3 Serial Port Connection

From the Personal Computer:

```
DSR (Pin 6)  must be marking
CTS (Pin 5)  must be marking
```

To the Personal Computer:

```
DTR (Pin 20) set to marking at initialization time
RTS (Pin 4)  set to marking at initialization time
```

If you use XON and XOFF to control the amount of data received by the serial device's input buffer, you can connect Pin 4 to Pins 5 and 6 at the Personal Computer end. If you use DSR (Pin 6) to control data transmission, you can connect Pin 4 to Pin 5 at the Personal Computer end. XON is an ASCII CTRL-Q or 011H; XOFF is an ASCII CTRL-S, or 013H. Data is transmitted from the Personal Computer on Pin 2; data is received at the Personal Computer on Pin 3.

4.6 The DDT86 (Dynamic Debugging Tool) Command

Syntax:

```
DDT86 { filespec }
```

The DDT-86 utility allows you to monitor and test programs developed for the 8086 and 8088 microprocessors.

DDT-86 responds to the single-letter commands, listed in Table 4-1.

Table 4-1. DDT-86 Single-Letter Commands

Command		Meaning
A	(Assemble)	enters assembly language statements.
B	(Block Cmp)	compares blocks of memory.
D	(Display)	displays memory in hexadecimal and ASCII.
E	(Execution)	loads program for execution.
F	(Fill)	fills memory block.
G	(Go)	begins execution.
H	(Hex)	hexadecimal sum and difference.
I	(Input)	sets up input command line.
L	(List)	lists memory in mnemonic form.
M	(Move)	moves memory block.
QI	(Query I)	reads port.
QO	(Query O)	writes port.
R	(Read)	reads diskette file to memory.
S	(Set)	sets memory values.
SR	(Search)	searches for string.
T	(Trace)	traces program execution.
U	(Untrace)	monitors execution without trace.
V	(Verify)	shows memory layout after diskette read.
W	(Write)	writes content of memory block to disk.
X	(Examine)	examines and modifies CPU registers.

The overall operation of DDT-86, along with each command, is described in detail in Section 11 of the Concurrent CP/M-86 Operating System Programmer's Guide.

If the file specification is not included, DDT-86 is loaded into user memory without a test program. You must not use the DDT-86 commands G, T, or U until you have first loaded a test program. The test program is usually loaded using E command.

If the file specification is included, both DDT-86 and the test program file specified by filespec are loaded into user memory. Use G, T, or U to begin execution of the test program under supervision of DDT-86.

If the filetype is omitted from the file specification, a filetype of CMD is assumed.

DDT-86 cannot directly load test programs in Hexadecimal (H86) format. You must first convert to command file form (CMD) using the GENCMD utility.

To exit from DDT-86, press CTRL-C.

Passwords

DDT-86 does not support passwords. If your files require passwords, be sure they all have the same one, and set the default password to this password.

Examples:

```
0A>DDT86
```

The DDT-86 utility is loaded from drive A to user memory. DDT-86 displays the - prompt when it is ready to accept commands.

```
0A>B:DDT86  TEST.CMD
```

The DDT-86 utility is loaded from drive B to user memory. The program file TEST.CMD is then loaded to User Memory from drive A. DDT-86 displays the address at which the file was loaded and the - prompt.

4.7 The DIR (Directory) Command

Syntax:

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

DIR displays the names of files stored on the specified diskette. If you do not specify a particular diskette, DIR returns the directory of the default diskette.

DIR only displays the filenames in the default user area of the diskette directory. Remember that the default user area is indicated by the number, 0 to 15, appearing to the left of the drive letter (A, B, C, D, or M) in the system prompt. DIR alone does not display files with the system attribute, [SYS]. Use DIR with the SYS option to display files with the SYS attribute turned on.

DIR needs no command tail, but accepts either a drive specification or a filename or both. The filename can use the * and ? to indicate wildcards. In response to DIR without a command tail, Concurrent CP/M-86 displays the filenames on the default drive, in the default user area, as shown below:

```
0A>DIR
Directory for User  0:
A: DIR      CMD : PIP      CMD : SUBMIT   CMD : ERAQ     CMD
A: ED       CMD : ASM86    CMD : DDT86    CMD : ABORT    CMD
A: STAT     CMD : TOD      CMD : DSKRESET CMD : CONSOLE  CMD
A: TEST1    DAT : TEST2    DAT :
A: FORMS    LIB : INDEX    LIB
```

If no command tail is given, as in the above example, Concurrent CP/M-86 treats the command as if you specified an ambiguous *.* filename and displays all the filenames on the default drive in the default user area. DIR with a drive specification also works as if you entered *.* , but Concurrent CP/M-86 displays the directory of the requested drive, still in the same user area, as shown in the following example:

```
0A>DIR B:
Directory for User  0:
B: DOCFILE1 TXT : DOCFILE2 TXT : DOCFILE3 TXT : NEWPROG  A86
B: NEWPROG  BAK : ADDPROG  A86 : DOCFILE1 BAK : CHECKPRG A86
B: DOCFILE3 BAK : CHECKPRG BAK : ADDPROG  BAK : DOCFILE2 BAK
```

DIR with a file specification searches the directory for the requested filename and displays it if it exists. Otherwise, Concurrent CP/M-86 returns a File Not Found error message. DIR with an ambiguous filename displays any directory filename that matches, as shown in the following examples:

```
0A>DIR *.CMD
Directory for User  0:
A: DIR      CMD : PIP      CMD : SUBMIT   CMD : ERAQ     CMD
A: ED       CMD : ASM86    CMD : DDT86    CMD :
A: STAT     CMD : TOD      CMD : DSKRESET CMD

0A>DIR B:DOCFILE?.*
Directory for User  0:
B: DOCFILE1 TXT : DOCFILE2 TXT : DOCFILE3 TXT : DOCFILE1 BAK
B: DOCFILE3 BAK : DOCFILE2 BAK
```

Use DIR to check the contents of your diskette and to verify that any file operations, such as renaming, erasing, or moving were performed correctly. If your diskette directory is longer than your console screen can display at one time, you might need to press a CTRL-S or Scroll-Lock to temporarily halt the console display before the top scrolls past you. To continue the display, press a CTRL-Q. If you strike a key other than CTRL-S or Scroll-Lock during the directory display, DIR aborts immediately.

You might want to make a printed copy of a diskette directory to keep with the diskette. To do this, press a CTRL-P before entering the DIR command. Concurrent CP/M-86 then lists the directory at the printer as well as at the console. Press another CTRL-P to stop all console activity echoing to the printer.

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

```
0A>DIR *.CMD [SYS]
```

The system file attribute (SYS) is intended for command files with their system attribute set. The SYS attribute allows access to the command files from any valid drive or user number even though they are located in user 0 on the system drive. File attributes are assigned using the SET utility (described in Section 4.20).

If you do not use the [SYS] option, and system files do exist on the directory, DIR displays a System Files Exist message. In the following example, the A drive is the system drive, and it contains a number of utility programs.

```
0A>DIR
Directory for User  0:
A: FORMS    LIB : TEST1    DAT : TEST2    DAT : INDEX    LIB
System Files Exist
```

With the [SYS] option, DIR also displays the system files.

```
0A>DIR [SYS]
Directory for User  0:
A: FORMS    LIB : TEST1    DAT : TEST2    DAT : INDEX    LIB
A: DIR      CMD : PIP      CMD : SUBMIT   CMD : ERAQ     CMD
A: ED       CMD : ASM86    CMD : DDT86    CMD : LOAD     PRL
A: STAT     CMD : TOD      CMD : CONSOLE  CMD
A: ABORT    CMD
```

The system files are not normally displayed, so they can be viewed as built-in to the Concurrent CP/M-86 system.

```
0A>DIR *.A86 [G8]
```

Use the G option to get and display the directory from another user number. Follow G with the number of the user area you want to display, from 0 to 15. In the following example, the DIR command displays all the files of type A86 in user number 8 on drive A.

```
0A>DIR *.A86[G8]
Directory for User  8:
A: PROGRAM1 A86 : PROGRAM2 A86 : TEST     A86
```

Multiple DIR commands can be given in one line. Each command must be separated from previous commands with a comma or space. In the following example, DIR first displays the .A86 files, and then the .LIB files from user 0 on Drive A. The SYS and G options effect all filenames in the command line.

```
0A>DIR *.A86, *.LIB
```

4.8 The DSKMAINT Command

Syntax:

```
        DSKMAINT
```

The DSKMAINT command is a multifunction menu-driven command that performs the most commonly used diskette maintenance operations: formatting new or reusable diskettes, copying the contents of one diskette to another, and verifying the integrity of the data on diskettes.

DSKMAINT does not copy individual files across source and destination diskettes. Diskettes are copied track-for-track, with each track written, verified, and reread as a final check.

4.8.1 Single- and Double-Sided Diskettes

If you want to copy an entire diskette with DSKMAINT, you must copy between two diskettes of the same type: two single-sided diskettes, or two double-sided diskettes. Look at the box of diskettes or at the label on an individual diskette to determine if it is single- or double-sided.

To make a double-sided copy of a single-sided diskette, follow the steps below to format your double-sided diskette. Then use PIP to copy individual files between the two dissimilar diskettes. The format option to DSKMAINT copies the necessary system loader tracks onto your new diskette.

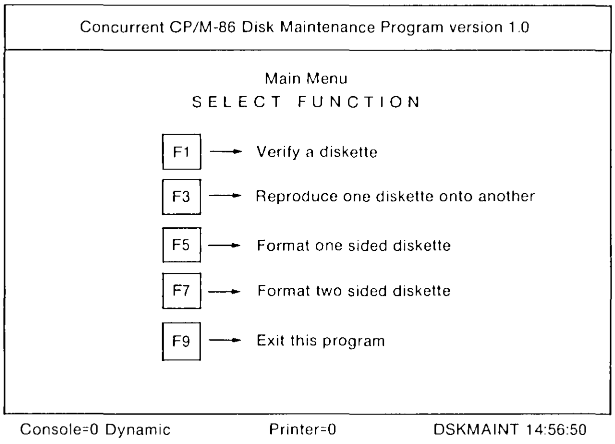
4.8.2 Formatting Diskettes

Before you call DSKMAINT, be sure your current virtual console is in Dynamic mode. Set it to Dynamic mode with VCMODE.

When you give the command:

```
        0A>dskmaint
```

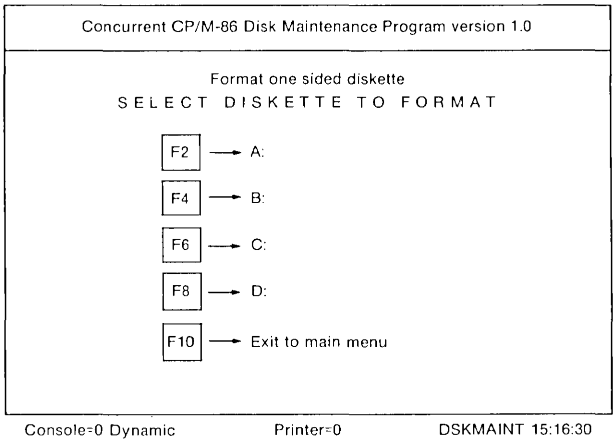
the screen in Figure 4-3 appears. At this point, you can remove diskettes from the drives. Place your unformatted diskette in the appropriate drive.



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

                    Main Menu
              S E L E C T   F U N C T I O N

              F1  --> Verify a diskette
              F3  --> Reproduce one diskette onto another
              F5  --> Format one sided diskette
              F7  --> Format two sided diskette
              F9  --> Exit this program

Console=0 Dynamic        Printer=0        DSKMAINT 14:56:50
```

Figure 4-3. DSKMAINT Main Menu

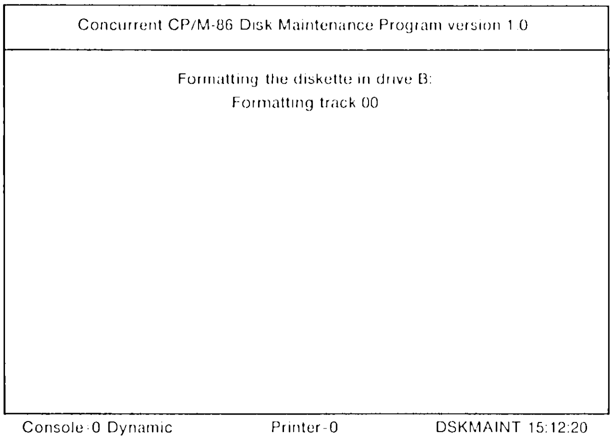
This is the DSKMAINT main menu. Press key F5 to format a single-sided diskette, or F7 for a double-sided diskette. Pressing F5 shows the screen in Figure 4-4. If you pressed F7, the only difference is in the title, which reads "Formats two sided diskette".



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

               Format one sided diskette
        S E L E C T   D I S K E T T E   T O   F O R M A T

              F2  --> A:
              F4  --> B:
              F6  --> C:
              F8  --> D:
              F10 --> Exit to main menu

Console=0 Dynamic        Printer=0        DSKMAINT 15:16:30
```

Figure 4-4. Format One Sided Diskette

Press the key that specifies the drive containing your disk; F2 for drive A, F4 for drive B, and so forth. This screen disappears and the screen in Figure 4-5 appears:



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

Formatting the diskette in drive B:
Formatting track 00

Console-0 Dynamic          Printer-0          DSKMAINT 15:12:20
```

Figure 4-5. Formatting the Diskette

When DSKMAINT finishes formatting your diskette track-by-track, it automatically writes the Concurrent CP/M-86 system loader information to it, then verifies the diskette contents track-by-track, counting down the tracks as it works. The Formatting track n message changes to Verifying track n. When the verification is through the screen returns to the main menu.

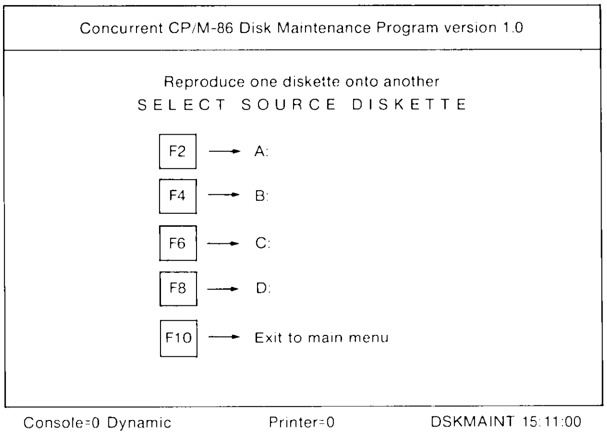
4.8.3 Reproducing Diskettes

Before you call DSKMAINT, be sure your current virtual console is in Dynamic mode. Set it to Dynamic mode with VCMODE.

When you give the command:

```
0A>dskmaint
```

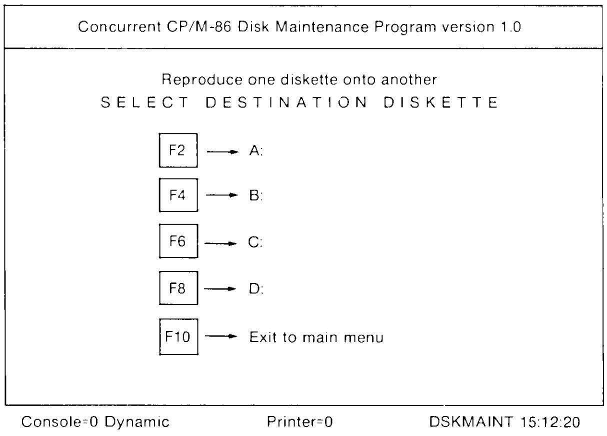
the screen in Figure 4-3, called the main menu, appears. At this point replace the old diskettes with the diskettes you want to copy across. Press F3 to copy diskettes. The screen in Figure 4-6 appears:



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

              Reproduce one diskette onto another
              S E L E C T   S O U R C E   D I S K E T T E

                    F2  --> A:
                    F4  --> B:
                    F6  --> C:
                    F8  --> D:
                    F10 --> Exit to main menu

Console=0 Dynamic        Printer=0        DSKMAINT 15:11:00
```

Figure 4-6. Reproduce Diskette: Select Source Diskette

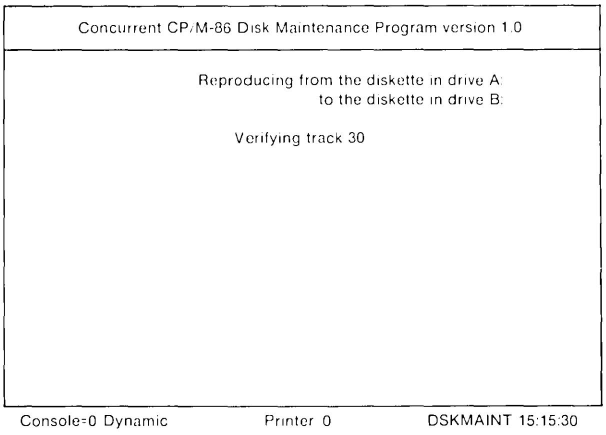
This screen asks you to specify which drive contains the source diskette with the original files to copy. Press the appropriate key; F2 for drive A, F4 for drive B, and so forth. This screen is replaced by the screen in Figure 4-7, which asks you to specify the drive containing the destination diskette. The destination diskette receives the copied files.



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

              Reproduce one diskette onto another
           S E L E C T   D E S T I N A T I O N   D I S K E T T E

                    F2  --> A:
                    F4  --> B:
                    F6  --> C:
                    F8  --> D:
                    F10 --> Exit to main menu

Console=0 Dynamic        Printer=0        DSKMAINT 15:12:20
```

Figure 4-7. Reproduce Diskette: Select Destination Diskette

As the copying operation takes place DSKMAINT informs you of its progress. The screen in Figure 4-8 tells you which disk is the source and which is the destination, noting the tracks as they are written, verified, and then reread to confirm proper operation.



```
Concurrent CP/M-86 Disk Maintenance Program version 1.0

                    Reproducing from the diskette in drive A:
                                 to the diskette in drive B:

                         Verifying track 30

Console=0 Dynamic           Printer 0           DSKMAINT 15:15:30
```

Figure 4-8. Reproducing the Diskette

After DSKMAINT finishes copying, the main menu returns to the screen.

4.8.4 DSKMAINT Error Messages

Before you call DSKMAINT, be sure your current virtual console is in Dynamic mode. Set it to Dynamic Mode with VCMODE.

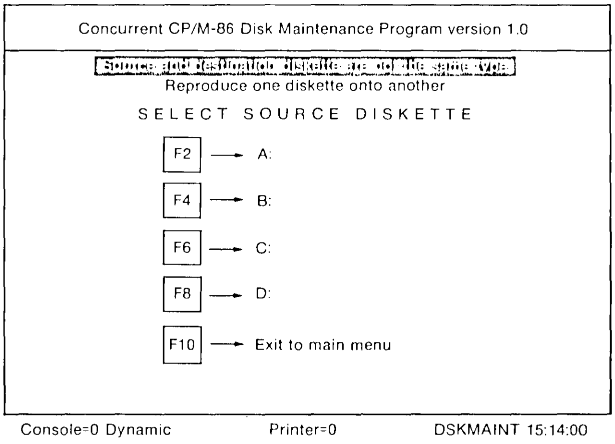
You might see an error message while performing a DSKMAINT operation; these messages appear in the Status Line at the bottom of your screen. This message is:

```
Disk Error  Accept/Retry/Ignore/Details?
```

Type A for Accept. DSKMAINT returns to the main menu and shows the BAD DISK TRY ANOTHER ONE error message printed in reverse video (black letters on a green strip) in the location shown in Figure 4-9. Remove the appropriate diskette, replace it with another, and the operation again.

If you have both double- and single-sided diskettes in your drives and try to copy across them, you obtain the following error message in the same location and format as the BAD DISK message above:

Source and destination diskette are not the same type



Concurrent CP/M-86 Disk Maintenance Program version 1.0

Source and destination diskette are not the same type

Reproduce one diskette onto another

SELECT SOURCE DISKETTE

Console=0 Dynamic Printer=0 DSKMAINT 15:14:00

Figure 4-9. Source and Destination Diskettes Not Same Type

4.9 The DSKRESET Command

Syntax:

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

The DSKRESET command enables the operator to change floppy diskettes or removable hard disks. The DSKRESET command with no command tail resets all the diskette drives. DSKRESET with specified drives in the command tail resets only those drives. After turning on the system, Concurrent CP/M-86 automatically resets all the drives. It is important to reset a drive with the DSKRESET command before changing the diskette in that drive. If the disk reset is successful, it means no other process is using the diskette and it can be safely changed.

The DSKRESET command checks the drive for any open files. If DSKRESET does not find any open files, it resets the drive.

When changing diskettes, you must reset the drive to enable writing to the new diskette. If you change a diskette and forget to reset the drive and then try to write data to that diskette, Concurrent CP/M-86 notices that the drive has not been reset, automatically sets that drive to Read-Only, and does not write data to any files on that diskette. Therefore, if you forget to reset the drive when you change a diskette, it is possible to lose an entire edit (changes that you are making to a file).

Example:

The following example demonstrates what happens if DSKRESET finds an open file on a drive.

```
0A>DSKRESET B:
Disk reset denied, Drive B: Console 2  Program  PIP
```

Concurrent CP/M-86 denied the request to reset the diskette in drive B because PIP, which was initiated at console 2, has an open file on drive B.

Note: It is extremely important to execute the DSKRESET command before and after changing a diskette. If a diskette is removed from a drive while a program is executing, the data in the open file on that disk might be irrevocably damaged. If you then insert another diskette, Concurrent CP/M-86 might write data onto this diskette, possibly overwriting files. Issuing the DSKRESET command after replacing the diskette logs in the new diskette so Concurrent CP/M-86 can read from its directory.

4.10 The ED (Character File Editor) Command

Syntax:

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

The ED utility lets you create or edit the diskette file named in the input file specification.

ED is a line-oriented context editor. This means that you create and change character files line-by-line, or by referencing individual characters in a line.

The ED utility uses a portion of user memory as the active text buffer where you add, delete, or alter the characters in the file. You use the A command to read all or a portion of the file into the buffer. You use the W or E command to write all or a portion of the characters from the buffer back to the file.

An imaginary character pointer, called CP, is at the beginning of the buffer, between two characters in the buffer, or at the end of the buffer.

You interact with the ED utility in either command or insert mode. ED displays the * prompt on the screen when ED is in command mode. When the * appears, you can enter the single-letter command that reads text from the buffer, moves the CP, or changes the ED mode of operation.

Table 4-2. ED Single-letter Commands

Command	Meaning
a (Append)	loads lines of text to the buffer.
b (Begin/Bottom)	moves CP to beginning or bottom of the buffer.
c (Character)	moves CP to right or left by characters.
d (Delete)	deletes characters to the right or left of the CP.
e (End)	ends edit session and writes buffer.
f (Find)	moves CP to a character sequence.
h (Head)	writes buffer, moves to beginning of the file.
i (Insert)	changes from command to insert mode.

Table 4-2. (continued)

Command	Meaning
j (Juxtapose)	places characters next to each other.
k (Kill Lines)	removes full lines above or below the CP.
l (Move Lines)	moves CP up or down by full lines.
m (Macro)	repetitively executes a command group.
n (Next)	finds next occurrence, automatically fills buffer.
o (Original)	discards current edit, restarts with original.
p (Page)	moves CP up or down by 23 full lines.
q (Quit)	discards current edit, no changes to the file.
r (Read)	reads LIB file to transfer text.
s (Substitute)	substitutes one character sequence for another.
t (Type)	types full lines on the display.
u (Upper Case)	translates all input to upper case.
v (Verify)	sets line number mode, or shows buffer space.
w (Write)	writes lines of text from the buffer.
x (Transfer)	transfers lines to or from a temporary file.
z (Sleep)	delays command execution.

Section 5 gives a detailed description of the overall operation of the ED utility and the use of each command.

Include the second file specification only if the file named by the first file specification is already present, and you do not want the original file replaced. The file named by the second file specification receives the altered text from the first file, which remains unchanged.

If the second file specification contains only the drive specification, the second filename and filetype become the same as the first filename and filetype.

If the file given by the first file specification is not present, the ED utility creates the file and writes the following message:

```
NEW FILE
```

If the second filespec is omitted, the original file is preserved by renaming the filetype to BAK before it is replaced. If you issue an ED command line that contains a filespec with filetype BAK, ED creates and saves your new edited version of the BAK file, but ED deletes your source file, leaving no back-up. If you want to save the original BAK file, use the REN command first to change the filetype from BAK, so that ED can rename it to BAK.

If you include the optional second filespec and give it the same name as the first filespec, ED again creates and saves your new edited version of the output filespec, but has to delete the original input filespec because it has the same name as the output file. You cannot, of course, have two files with the same name in the same user number on the same drive.

If the file given by the first filespec is already present, you must issue the A command to read portions of the file to the buffer. If the size of the file does not exceed the size of the buffer, the command

```
#a
```

reads the entire file to the buffer.

The i (Insert) command places the ED utility in insert mode. In this mode, any characters you type are stored in sequence in the buffer starting at the current CP.

Any single-letter commands typed in insert mode are not interpreted as commands, but are simply stored in the buffer. You return from insert mode to command mode by typing CTRL-Z.

The single-letter commands are usually typed in lower-case. The commands that must be followed by a character sequence end with CTRL-Z if they are to be followed by another command letter.

Any single-letter command typed in upper-case tells ED to internally translate to upper-case all characters up to the CTRL-Z that ends the command.

When enabled, line numbers that appear on the left of the screen take the form,

```
nnnnn:
```

where nnnnn is a number in the range 1 through 65535. Line numbers are displayed for your reference and are not contained in either the buffer or the character file. The screen line starts with

```
        :
```

when the CP is at the beginning or end of the buffer.

Examples:

```
        0A>ED MYPROG.A86
```

If not already present, this command line creates the file MYPROG.A86 on drive A. The command prompt,

```
        :*
```

appears on the screen. This tells you that the CP is at the beginning of the buffer. If the file is already present, issue the command:

```
        :*#a
```

to fill the buffer. Then type the command:

```
        :*0p
```

to fill the screen with the first 23 lines of the buffer. Type the command:

```
        :*e
```

to stop the ED utility when you are finished changing the character file. The ED utility leaves the original file unchanged as MYPROG.BAK and the altered file as MYPROG.A86.

```
        0A>ED MYPROG.A86  B:NEWPROG.A86
```

The original file is MYPROG.A86 on the default drive A. The original file remains unchanged when the ED utility finishes, with the altered file stored as NEWPROG.A86 on drive B.

```
        0A>B:ED MYPROG.A86  B:
```

The ED.CMD file must be on drive B. The original file is MYPROG.A86 located on Drive A. It remains unchanged, with the altered program stored on drive B as MYPROG.A86.

4.11 The ERA (Erase) Command

Syntax:

```
ERA filespec
```

The ERA command removes one or more files from the directory of a diskette. Wildcard characters are accepted in the command tail. Directory and data space are automatically reclaimed for later use by another file.

Use the ERA command with care, because all files that satisfy the file specification are removed from the diskette directory.

Command lines that take the form:

```
ERA {d:}*.*
```

require your acknowledgment, because they reclaim all file space. You see the following message:

```
Confirm delete all user files (Y/N)?
```

Respond with y if you want to remove all files, and n if you want to avoid erasing any files.

You see the message:

```
NO FILE
```

on the screen if no files match the file specification.

Examples:

```
0A>ERA X.A86
```

This command removes the file X.A86 from the diskette in drive A.

```
0A>ERA *.PRN
```

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

```
0B>ERA A:MY*.*
```

Each file on drive A with a filename that begins with MY is removed from the diskette.

```
0A>ERA B:*.*
```

All files on drive B are removed from the diskette. To complete the operation, you must respond with a y when the ERA command displays the following message:

```
Confirm delete all user files (Y/N)?
```

4.12 The ERAQ (Erase with Query) Command

Syntax:

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

The ERAQ command behaves exactly like the ERA command, with the addition of a query before each erasure.

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

Example:

```
        1A>ERAQ C:*.CMD

        C:ASM86        CMD ?y
        C:FUNCTION     CMD ?n
        C:SDIR         CMD ?y
        C:DSKRESET     CMD ?y
        1A>
```

In the example, the user instructs ERAQ to delete all the files except C:FUNCTION.CMD.

4.13 The FUNCTION (User-Programmable Function Keys) Command

Syntax:

```
FUNCTION [filename.]
FUNCTION [filename.typ]
```

FUNCTION is the command that lets you specify functions for the twenty Programmable Function Keys. You can change their programming, or simply view the current commands generated by the PFKs.

FUNCTION operates in one of two modes, specified by the command tail entered with FUNCTION. If you supply no command tail, FUNCTION displays an interactive menu for you to select from. If you supply a filespec in the command tail, FUNCTION takes its input from the specified file.

Concurrent CP/M-86 comes up with commands programmed into the ten F1-F10 function keys at the left of the keyboard--that is, Concurrent CP/M-86 finds a start-up file that specifies commands for these keys when the operating system is loaded. These commands are:

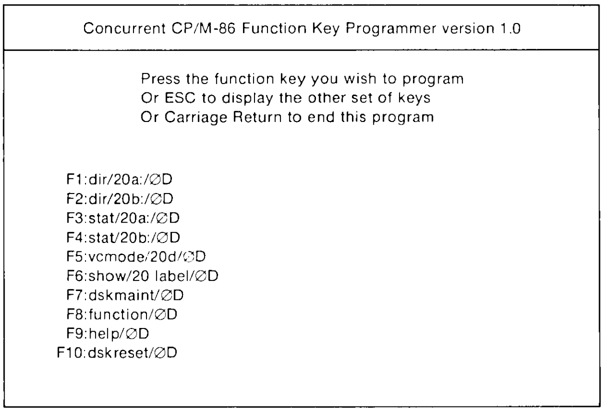
```
F1:  SDIR\0D
F2:  SDIR
F3:  STAT *.*\0D
F4:  STAT
F5:  VCMODE D\0D
F6:  VCMODE B\0D
F7:  DSKMAINT\0D
F8:  FUNCTION
F9:  HELP\0D
F10: DSKRESET\0D
```

The commands terminated with \0D indicate that these functions include carriage returns. When you press the key, Concurrent CP/M-86 acts as if you had given the command with a carriage return. The commands without \0D can take further specifications or parameters you supply. Terminate them with a carriage return and Concurrent CP/M-86 carries out the command.

As noted earlier, these commands are programmed by a start-up file (N.SUP) which invokes FUNCTION with the filespec DATA.PFK as the command tail. The command in this file is executed when Concurrent CP/M-86 starts up. You can change this start-up file, replace it, or eliminate it altogether and still retain access to this set of programmed commands. Pressing the shift key and any of the ten function keys produces the command.

4.13.1 Using the FUNCTION Menu

When you first call FUNCTION, the menu in Figure 4-10 appears.

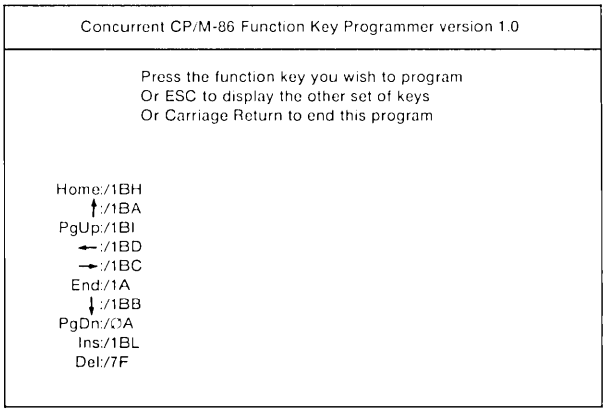


```
Concurrent CP/M-86 Function Key Programmer version 1.0

          Press the function key you wish to program
          Or ESC to display the other set of keys
          Or Carriage Return to end this program


 F1:dir/20a:/∅D
 F2:dir/20b:/∅D
 F3:stat/20a:/∅D
 F4:stat/20b:/∅D
 F5:vcmode/20d/∅D
 F6:show/20 label/∅D
 F7:dskmaint/∅D
 F8:function/∅D
 F9:help/∅D
F10:dskreset/∅D
```

Figure 4-10. Start-up FUNCTION Menu for F1-F10

The string of Concurrent CP/M-86 commands and symbols following F-number: is the programmed function. As the menu says, press any key F1-F10 to change the programmed value. Concurrent CP/M-86 revises the screen and prompts you to enter the new command string. You cannot edit the command strings as you enter them. If you make a mistake, terminate the string with \00 or CTRL-@ and start from the main menu again.

If you want to alter or inspect the other PFKs, press ESC. The screen in Figure 4-11 appears. Pressing ESC from either display switches you to the other. Enter a carriage return (↵) to return to the system command level.



```
              Concurrent CP/M-86 Function Key Programmer version 1.0

                    Press the function key you wish to program
                    Or ESC to display the other set of keys
                    Or Carriage Return to end this program


    Home:/1BH
       ↑:/1BA
    PgUp:/1BI
       ←:/1BD
       →:/1BC
     End:/1A
       ↓:/1BB
    PgDn:/0A
     Ins:/1BL
     Del:/7F
```

Figure 4-11. Start-up FUNCTION Menu for Keypad Keys

4.13.2 Nonprinting Characters

You can enter other characters besides the ASCII character set. You can specify absolute hexadecimal values so all possible control characters and 8-bit character codes can be used.

Enter absolute hexadecimal numbers as follows: first press the backslash key, indicating that the next 2 keystrokes are hexadecimal digits. If the next 2 digits are legal hexadecimal digits (0 through F), FUNCTION assembles them into a single byte and puts them into the PFK string for this key.

If you enter a 2 digit hexadecimal code of 5C, less than 21, or greater than 7E (representing nonprinting ASCII characters), then this number appears on the screen. On the other hand, if the hexadecimal digits represent a printing ASCII character (hexadecimal numbers 21 through 7E, with the exception of the backslash character \, 5C), then the appropriate character appears on your screen.

This process works in the other direction, too. If you enter a nonprinting character such as carriage return, you see the hexadecimal equivalent of that character on your screen.

4.13.3 Terminating Input

When you are satisfied with the new command string and want to program it in, press CTRL while also pressing @. If you mistakenly enter more than 19 characters, FUNCTION terminates the input automatically and returns to the key menu you started from, either the function key menu or the number pad key menu.

4.13.4 Using FUNCTION With a Command File

If you enter a filespec with FUNCTION, Concurrent CP/M-86 opens this file and programs the function keys according to the command strings found within. These files must include a line for each function key to be programmed. Look at the DATA.PFK file to see how such files are constructed; use a text editor like ED to create your own customized version.

4.13.5 Specifying the Key to be Programmed

When you build a file of command strings to program into the Personal Computer function keys, specifying the key to be programmed is not nearly as straightforward as when you use the interactive menu. FUNCTION looks for certain symbols, called key specifiers, at the beginning of each command string. Look at the DATA.PFK file once more; you see key specifiers in each command string. These key specifiers are listed in Table 4-3.

Table 4-3. Function Key Identifiers

Symbol	Function Key	Symbol	Function Key
;	F1	G	Home
<	F2	H	Up arrow
=	F3	I	Page Up
>	F4	K	Left Arrow
?	F5	M	Right Arrow
@	F6	O	End
A	F7	P	Down Arrow
B	F8	Q	Page Down
C	F9	R	Ins
D	F10	S	Del

For example, suppose you wanted to program the function keys F1 and F2 to invoke the system commands SDIR and SDIR B: respectively. You can program the keys using the FUNCTION menu, but you have to do so every time you loaded Concurrent CP/M-86. Entering these two lines in a start up file of PFK commands and saving that file lets Concurrent CP/M-86 program these two keys for you automatically upon start up.

```
;sdir\0D\00
<sdir b:\0D\00
```

As you can see from Table 4-3, the semicolon in the first line specifies that the F1 key is programmed with the following command string. The < symbol on the second line indicates that the F2 key is programmed. The \0D in each line represents the ASCII code (in hexadecimal format) for carriage return. As in the interactive mode, \00 (an ASCII null) indicates the end of the string.

You can put comments in each line to remind yourself what your intent is to program a particular command string. Comments can go at the beginning of the line preceding the key specifier, or at the end of the line following either the \00 or the 19th character. Comments preceding the key specifier cannot include any key specifiers. The following two lines have the same effect as the preceding lines, but they include comments:

```
This is the line to get sdir;SDIR\0D\00
<SDIR B:\0D\00This is the line to get sdir b:
```

Once you complete writing your command file and check it for errors, you can return to the system command level and call this command file to program the specified keys. If your file is named PFK.PFK, for example, give the command:

```
0A>FUNCTION PFK.PFK
```

FUNCTION reads the specified file, programs the keys accordingly, and returns to system command level.

4.14 The GENCMD (Generate CMD File) Command

Syntax:

```
GENCMD filespec {8080 CODE[An,Bn,Mn,Xn] DATA[An,Bn,Mn,Xn]
STACK[An,Bn,Mn,Xn] EXTRA[An,Bn,Mn,Xn]
```

The GENCMD utility produces an executable CMD file that can be used as a utility command. The input to GENCMD is an H86 file produced by ASM-86 or the Intel OH86 utility. The operation of GENCMD is described in detail in Section 4.2 of the Concurrent CP/M-86 Operating System Programmer's Guide. Included in the CMD file is a header containing information used at load time to determine memory requirements and segment register initialization.

An optional parameter list follows the file specification. In the parameter list, n represents a hexadecimal value up to four digits long.

The parameter list consists of up to five keywords with their corresponding list of values. The keywords are:

8080 CODE DATA STACK EXTRA

The keyword 8080 identifies the CMD file as an 8080 Memory Model where segment registers are initialized to the same value. The remaining keywords define segment groups which have specific memory requirements. The values that define the memory requirements are separated by commas and enclosed in square brackets ([]) following each keyword. The bracketed keywords and related values must be separated from other keywords by at least one blank.

The values included in brackets are defined below, where n represents a hexadecimal constant of from one to four digits. The value n represents a paragraph value, where each paragraph is 16 bytes long. The paragraph value corresponds to the byte value n * 16, or hhhh0 in hexadecimal.

An	Load Group at Absolute Location n
Bn	Begin Group at address n in the Hexadecimal File
Mn	The Group Requires a Minimum of n * 16 Bytes
Xn	The Group Can Address up to n * 16 Bytes

Use the 8080 keyword for programs from 8-bit microprocessors that are a direct conversion to Concurrent CP/M-86 so that the program loads into an area with overlapping code and data segments. The code segment in the program must begin at location 100H.

Use An for any group that must be loaded at an absolute location in memory. Do not use an A value in the command tail unless you know that the requested absolute area will be available when the program runs.

Use Bn when your input Hex file does not contain information that identifies the segment groups. This value is not necessary when your H86 file is the output from the Digital Research ASM-86 Assembler, unless the ASM-86 parameter FI was included.

Use the Mn value when you include a data segment that has an uninitialized data area at the end of the segment.

Use Xn when your program can use a larger data area, if available, than the minimum given by Mn.

Passwords

GENCMD does not support passwords. If your H86 files require passwords, be sure they all have the same one, and set the default password to this password.

Examples:

```
        0A>GENCMD MYFILE
```

The file MYPROG.H86 is read from drive A. The output file MYPROG.CMD is written back to drive A. The input H86 file includes information that marks the program as operating with a particular memory model.

```
        0B>GENCMD MYFILE  CODE[A40]  DATA[M30,XFFF]
```

The file MYFILE.H86 is read from drive B. The MYFILE.CMD output file is written to drive B. The code group must be loaded at location 400 hexadecimal. The data group requires a minimum of 300 hexadecimal bytes, but if available, the program can use up to FFF0 bytes.

4.15 The HELP (Help) Command

Syntax:

```
HELP {topic} {subtopic1 subtopic2 ... subtopic8}{[P]}
```

The HELP command provides summarized information for all of the Concurrent CP/M-86 commands described in this manual. 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 additional topics. 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. You can continue to specify topics for additional information, or simply press the ENTER key to return to the Concurrent CP/M-86 system prompt.

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

HELP with the [P] option prevents the screen display from stopping every 23 lines.

If you specify a password for the HELP.HLP file, the command fails.

Examples:

```
0A>HELP
```

The above command displays a list of topics for which help is available.

```
0A>HELP STAT
```

The above command displays general information about the STAT command. It also displays any available subtopics.

```
0A>HELP STAT OPTIONS
```

The above command includes the subtopic options. In response, HELP displays information about options associated with the STAT command.

```
0A>HELP ED
```

The above command displays general information about the ED utility.

```
HELP>ED COMMANDS
```

The above example shows how to enter a topic and subtopic following the program's internal prompt, HELP>.

Note: you must type the topic AND the subtopic, otherwise the HELP program cannot know which main topic you are referencing.

4.16 The PIP (Peripheral Interchange Program--Copy File) Command

Syntax:

```
PIP dest-file{[Gn]}|dev=src-file{[options]}|dev{[options]}
```

The PIP utility copies one or more files from one diskette 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 diskette to the printer or other auxiliary logical output device. PIP can create a file on diskette from input from the console or other logical input device. PIP can transfer data from a logical input device to a logical output device, hence the name Peripheral Interchange Program.

Note: PIP can copy from a single-sided diskette to a double-sided diskette, or vice-versa. However, before attempting to read from or write to a diskette of a different format, you must log in the new diskette with a DSKRESET immediately following the system prompt. If you forget to log in the diskette, PIP reads from the source diskette improperly.

4.16.1 Single File Copy

Syntax:

```
PIP d:{[Gn]} = source-filespec{[options]}
PIP dest-filespec{[Gn]} = d:{[options]}
PIP dest-filespec{[Gn]} = source-filespec{[options]}
```

The first form shows the simplest way to copy a file. PIP looks for the file named by source-filespec on the default or optionally specified drive. PIP copies the file to the drive specified by d: and gives it the same name as source-filespec. If you want, 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 source file specification (source-filespec). See Section 4.16.6 for a description of PIP options.

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 same name as 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 source-filespec, copies it to the location specified in dest-filespec, and gives it the name indicated by dest-filespec.

Before you start PIP, be sure that you have enough free space in kilobytes on your destination diskette to hold the entire file or files that you are copying. Even if you are replacing an old copy on the destination diskette with a new copy, PIP still needs enough room for the new copy before it deletes the old copy. (See Section 4.22, the STAT Command.)

Data is first copied to a temporary file to ensure that the entire data file can be constructed within the space available on the diskette. 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 (SYS, DIR, R/W, R/O) are transferred with the files.

If the existing destination file is set to Read-Only (R/O), PIP asks you if you want to delete it. Answer Y or N. Use the W option to write over Read-Only files.

You can include PIP options following each source name (see Section 4.16.6, PIP Options). There is one valid option ([Gn]--go to user number n) for the destination file specification. Options are enclosed in square brackets. Several options can be included for the source files. They can be packed together or separated by spaces. Options can verify that a file was 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 upper-case, and much more.

<u>Examples:</u>

```
0A>PIP B:=A:oldfile.dat

0A>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 source or destination 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 Section 4.16.6, PIP options, and Section 4.27, the USER utility.)

The second short form produces exactly the same result as the first one. PIP simply looks for the file oldfile.dat on drive A, the drive specified as the source.

```
        0A>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.

```
        0A>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 Concurrent CP/M-86 automatically starts out in user 0-the default user number) to the same drive and user number. This effectively gives you two copies of the same file on one drive and user number, each with a different name.

```
        0A>PIP B:PROGRAM.BAK = A:PROGRAM.DAT[G1]
```

The command above copies the file PROGRAM.DAT from user 1 on drive A to the currently selected user number on drive B (usually user 0), and renames the filetype on drive B to BAK.

```
        0B>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 drive and user number, in this case drive B.

4.16.2 Multiple File Copy

Syntax:

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

When you use a wildcard in the source specification, PIP copies qualifying files one-by-one to the destination drive, retaining the original name of each file. PIP displays the message COPYING followed by each file name 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:

```
        0A>PIP B:=A:*.CMD
```

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

```
        0A>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 diskette. Note, however, that this command does not copy the Concurrent CP/M-86 system from the system tracks. Use DSKMAINT to copy the system tracks.

```
        0A>PIP B:=A:PROG????.*
```

The above command causes PIP to copy all files beginning with PROG and having any filetype at all from drive A to drive B.

```
        0A>PIP B:[G1]=A:*.BAK
```

This command causes PIP to copy all the files with a filetype of BAK on drive A in the default user number (user 0, unless you changed the the user number with the USER command) to drive B in user 1. Remember that the DIR, TYPE, ERA, and other commands only access files in the same user number from which they were invoked. (See Section 4.27, the USER utility.)

4.16.3 Combining Files

Syntax:

```
        PIP dest-file{[Gn]}=src-file{[o]},file{[o]}{,file{[o]}...}
```

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.

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

```
        Gn  O  R  V  W
```

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

During character transfers, you can terminate a file concatenation operation by striking any key on your keyboard.

When concatenating files, PIP searches only 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, normally used to indicate the end of character files.

Examples:

```
0A>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 diskette in the default drive A.

```
0A>PIP B:X.A86  = Y.A86 , B:Z.A86
```

The file Y.A86 on drive A is joined with Z.A86 from drive B and placed in the temporary file X.$$$ on drive B. The file X.$$$ is renamed to X.A86 on drive B when PIP runs to successful completion.

4.16.4 Copy Files to and from Auxiliary Devices

Syntax:

```
PIP dest-filespec {[Gn]} = source-filespec {[o]}

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

This form is a special case of the PIP command line that lets you copy a file from a diskette to a device, from a device to a diskette, or from one device to another. The files must contain printable characters. Each peripheral device has a logical name that identifies a source device that can transmit data or a destination device that can receive data. A colon follows each logical device name so it cannot be confused with a filename. Strike any key to abort a copy operation that uses a logical device in the source or destination.

The logical device names are:

- CON: Console: the physical device assigned to CON:. When used as a source, usually the keyboard; when used as a destination, usually the screen.
- LST: The destination device assigned to LST:, usually the printer.

There are three device names that have special meaning:

- NUL: A source device used for paper tape readers that produces 40 nulls (hexadecimal zeros).
- EOF: A source device that produces a single CTRL-Z (the Concurrent CP/M-86 end-of-file mark).
- PRN: The printer device with tab expansion to every eighth column, line numbers, and page ejects every 60th line.

Examples:

```
0B>PIP PRN:=CON:,MYDATA.DAT
```

Characters are first read from the console input device, generally the keyboard, and sent directly to your printer device. You type a CTRL-Z character to tell 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.

```
0A>PIP B:FUNFILE.SUE = CON:
```

If CON: is assigned to input, 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.

```
0A>PIP LST:=CON:
```

If CON: is assigned as input, whatever you type at the keyboard is written to the list device, generally the printer. Terminate input with a CTRL-Z.

```
0A>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 8.

```
0A>PIP PRN:=B:DRAFT.TXT
```

The command above 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.

4.16.5 Multiple Command Mode

Syntax:

```
PIP
```

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 pushing only the carriage return key following the asterisk prompt. The empty command line tells PIP to discontinue operation and return to the Concurrent CP/M-86 system prompt.

Note: this form of PIP lets you change source diskettes of the same formats between commands. You cannot, however, change from a single-sided to a double-sided source diskette, or vice-versa. If you do, PIP reads the source diskette improperly. You cannot change the destination diskette at all. If you do, Concurrent CP/M-86 displays a Read-Only error Message.

Examples:

```
0A>PIP
*NEWFILE=FILE1,FILE2,FILE3
*APROG.CMD=BPROG.CMD
*A:=B:X.A86
*B:=*.*
 *
```

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 shown in the previous examples, where the command line is included in the command tail. PIP is not loaded into memory for each command.

4.16.6 Using Options With PIP

Options enable you to process your source file in special ways. You can expand tab characters, translate from upper- to lower-case, extract portions of your text, verify that the copy is correct, and much more.

The PIP options are listed below, 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 effects. The option must be enclosed in square brackets []. 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 square brackets [].

Table 4-4. PIP Options

Option	Meaning
A	To back up only the files that have been modified since the last back-up, use PIP with an ambiguous filename and the Archive option. PIP with the [A] option copies only the files that have been modified.
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 embedded in the source data.
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 in the range 0 to 15.
H	Hex data transfer. PIP checks all data for proper Intel hexadecimal file formats. 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.

Table 4-4. (continued)

Option	Meaning
L	Translate upper-case alphabetics in the source file to lower-case 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 1 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 (noncharacter and therefore nonprintable) files. PIP ignores any CTRL-Z ends-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 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.
R	Read system (SYS) files. Normally, PIP ignores files marked with the system attribute in the diskette 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.

Table 4-4. (continued)

Option	Meaning
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 lower-case alphabetic characters in the source file to upper-case in the destination file. This parameter follows the source device or filename.
V	Verify that data has been copied correctly. PIP compares the destination to the source data to ensure that the data has been written correctly. The destination must be a diskette file.
W	Write over files with R/O (Read-Only) attribute. Normally, if a PIP command tail includes an existing R/O file as a destination, PIP sends a query to the console to make sure you want to write over the existing file. When this parameter follows a source name, PIP overwrites the R/O 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:

```
0A>PIP NEWPROG.A86 =CODE.A86 [L], DATA.A86 [U]
```

This command constructs the file NEWPROG.A86 on drive A by joining the two files CODE.A86 and DATA.A86 from drive A. During the copy operation, CODE.A86 is translated to lower-case, while DATA.A86 is translated to upper-case.

```
0A>PIP CON:=WIDEFILE.A86 [D80]
```

This command writes the character file WIDEFILE.A86 from drive A to the console device, but deletes all characters following the 80th column position.

```
0A>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.

```
        0A>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.

```
        0B>PIP LST:=PROGRAM.A86 [NT8U]
```

This command writes the file PROGRAM.A86 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 lower-case letters to upper-case as the file is printed.

```
        0A>PIP PORTION.TXT=LETTER.TXT[SDear Sir^Z QSincerely^Z]
```

This command abstracts 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.

```
        0B>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 marked R/O (Read-Only). The R parameter tells PIP to read files from drive A marked with the SYS (System) attribute.

4.17 The PRINTER Command

Syntax:

```
PRINTER
PRINTER n
```

The printer command displays or sets the printer used by a particular console. Several consoles can share the same printer, but only one process can use the printer at a time. Concurrent CP/M-86 expects the first printer assigned to the system to be printer number 0. The second printer is printer 1, and so on. The printer number is specified by n. When the PRINTER command includes an n, PRINTER sets the list device to printer number n. When you enter PRINTER without the n option, the system returns the number of the printer currently assigned to your console.

Examples

```
0A>PRINTER
Printer Number = 1

0A>PRINTER 2
Printer Number = 2
```

4.18 The REN (Rename) Command

Syntax:

```
REN {d:}newname{.typ} = oldname{.typ}
```

The REN command lets you change the name of a file that is cataloged in the directory of a diskette.

The filename oldname identifies an existing file on the diskette. The filename newname is not in the directory of the diskette. The REN command changes the file named by oldname to the name given as newname.

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

If you omit the drive specifier, REN assumes the file to rename is on the default drive.

You can include a drive specification as a part of the newname. If both file specifications name a drive, it must be the same drive. If the file given by oldname does not exist, REN displays the following message on the screen:

```
No such file to rename
```

If the file given by newname is already present in the directory, REN displays the following message on the screen:

```
Not renamed: Newfile already exists, delete (Y/N)?
```

Examples:

```
0A>REN NEWASM.A86 =OLDFILE.A86
```

The file OLDFILE.A86 changes to NEWASM.A86 on drive A.

```
0B>REN A:X.PAS = Y.PLI
```

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

```
0A>REN B:NEWLIST=B:OLDLIST
```

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

```
0A>REN B:NEWLIST=OLDLIST
```

4.19 The SDIR Command

Syntax:

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

The following exceptions are allowed:

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

The SDIR utility is an enhanced combination of the DIR utility and the STAT utility. SDIR is equipped with all of the options needed to display Concurrent CP/M-86 files in a variety of ways. SDIR can search for files on any or all drives, in any or all user areas.

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

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

SDIR searches on single file specifications or on combinations of up to 10 file specifications. As explained in Section 2, a file specification consists of an optional drive specification and colon, followed by a filename, followed by an optional period and filetype, followed by any necessary semicolon and password. SDIR ignores passwords. The filename and filetype can include asterisks and/or question marks for wildcard searching. SDIR treats d: as if it were the ambiguous filespec, d:*.*.

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

```
0A>SDIR [HELP]
```

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

SDIR has three formats: the default or full format, the size format, and the short format.

4.91.1 SDIR Full Format

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

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

```
6E>sdir m:

Directory For Drive M:   User  6

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

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

SDIR displays the Read-Only or Read-Write, the SYS or DIR, the Archive, and the user-defined (F1, F2, F3, F4) attributes of a file. SDIR displays SYS if the System attribute of the file is on, and DIR if it is off. SDIR displays RO if the Read-Only attribute is on, and RW if it is off. SDIR displays the number 1, 2, 3, or 4 corresponding to the number of any user attributes that are on.

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

4.19.2 SDIR Size Format

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

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

4.19.3 SDIR Short Format

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

4.19.4 SDIR Options

The following table explains each of the SDIR options.

Table 4-5. SDIR Options

Command	Result
SDIR	displays all files on the default drive, in the default user area, in full format, sorted alphabetically. This command is the default display and is equivalent to SDIR [RO, RW, SYS, DIR, SORT, FULL, XFCB].
SDIR [SYS]	displays only the files that have the SYS attribute on.
SDIR [RO]	displays only the files that have Read-Only attribute on.

Table 4-5. (continued)

Command	Result
SDIR [DIR]	displays only the files that have the SYS attribute off.
SDIR [RW]	displays only the files that are set to Read-Write.
SDIR [XFCB]	displays all the files that have Extended File Control Blocks (XFCBs). See the SET command for a discussion of XFCBs. The SHORT option is ignored when used with the XFCB option.
SDIR [NONXFCB]	displays those files without Extended File Control Blocks. The SHORT option is ignored when used with the NONXFCB option.
SDIR [USER=n]	displays files under the user number specified by n.
SDIR [USER=ALL]	displays files under all the user numbers for the default drive.
SDIR [USER=(0,1,...15)]	displays files under the user numbers specified.
SDIR [DRIVE=d]	displays files on the drive specified by d. The drive specified must exist. DISK is also acceptable in place of DRIVE in all the DRIVE options.
SDIR [DRIVE=ALL]	displays files on all of the logged-in drives. A logged-in drive is a valid existing drive that has been accessed since the last DSKRESET.

Table 4-5. (continued)

Command	Result
SDIR [DRIVE=(A,B,C,...P)]	displays files on the drives specified. The specified drives must exist.
SDIR [FULL]	displays the files in full format. If no directory label exists for the drive, or the NONXFCB option was specified, the columns for date and time stamps are omitted, and two files are displayed per line. This is the default display.
SDIR [LENGTH=n]	displays n lines of filenames before inserting a table heading. n must be in the range between 5 and 65536.
SDIR [SIZE]	displays the diskette space in kilobytes allotted to the files on the default or specified drive.
SDIR [FF]	The form-feed option is used with the CTRL-P character to make hard copies of directories. It sends an initial form-feed to the printer. If the LENGTH option is also specified, a form-feed is issued every n lines.
SDIR [MESSAGE]	The message option is used when SDIR is searching for files on more than one drive and/or user area. Normally, SDIR does not print the names of the drives and users it is searching. With this option, SDIR displays the names of the specified drives and user areas and any files residing there. If there are no files in the specified locations, SDIR displays the File not found message.

Table 4-5. (continued)

Command	Result
SDIR [NOSORT]	SDIR normally sorts files alphabetically. SDIR [NOSORT] displays the files in the order it finds them in the directory. When the SHORT format is used, NOSORT is automatically set.
SDIR [EXCLUDE]	SDIR with the EXCLUDE option displays the files that do not match the files specified in the command line.
SDIR [HELP]	displays examples of various SDIR commands.
SDIR [SHORT]	displays files in four columns and excludes password and time stamping columns. NOSORT is automatically implemented with this option and the files are not listed alphabetically.

<u>Examples:</u>

The examples below illustrate some of the uses of the SDIR command. The following command line instructs SDIR to list all the System files of type PLI, CMD, and A86 on the system in the currently logged drives in any user area.

```
0A>SDIR [user=all,drive=all,sys] *.PLI *.CMD *.A86
```

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

```
0A>SDIR [drive=all user=all] TESTFILE.BOB
```

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

```
0A>SDIR [size,rw] D:
```

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

```
        0A>SDIR [xfcb] D:*.CMD
```

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

```
        0A>SDIR [short] A: B: C:
```

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

```
        0A>SDIR [exclude] *.CMD
```

4.20 The SET Command

Syntax:

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

The SET command initiates password protection and time stamping of files in the Concurrent CP/M-86 system. It also sets file and drive attributes, such as the Read Only, System, and user-definable attributes. This section discusses three major topics: password protection, time stamping, and setting file attributes.

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

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

4.20.1 Password Protection

The Concurrent CP/M-86 Operating System supports password protection of files. Passwords can be up to 8 characters long and are required for access to the file to which they are assigned. Passwords can be assigned to the Concurrent CP/M-86 commands present on your system (CMD files). When accessing a password-protected file or command, the password must be preceded by a semicolon and typed as part of the command or filename as shown in the following example.

```
0A>TYPE B:DOCUMENT.LAW;SECRET
0A>ERA;SECRET B:DOCUMENT.LAW;SECRET
```

In the first example above, the file DOCUMENT.LAW is protected by the password SECRET. The password must be included in the file specification to type the file. In the second example, both the Concurrent CP/M-86 ERA command (See Section 4.11) and the file DOCUMENT.LAW are protected by the password SECRET. Some Concurrent CP/M-86 commands prompt for the password if it is missing from the file specification. For instance, in the ERA command above, the password for B:DOCUMENT.LAW could have been left off as shown below.

```
0A>ERA B:DOCUMENT.LAW

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

0A>
```

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

4.20.2 Turning On Password Protection

SET controls password protection for individual files and for the directory or drive label. The optional directory or drive label is an entry in the directory. Just as a directory entry describes a file to Concurrent CP/M-86, the label contains information that describes special attributes of the diskette to the operating system. For example, the label tells Concurrent CP/M-86 whether or not time stamping and password protection are turned on for that diskette. You can give the label a name to help identify the data that is stored on a given diskette.

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

It is extremely important that you assign a password to the directory label if you intend to use the password protection features of Concurrent CP/M-86. Because of its power, it is useful to think of the directory label password as a kind of super-password. If you know the super-password, you can turn password protection off for the whole drive and thereby gain access to all files on the drive, even those which are protected by passwords you do not know.

```
0A>SET [PASSWORD=SECRET]
0A>SET [PASS=<cr>
```

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

```
0A>SET [PASSWORD=SECRET]

Label for drive A:

Directory        Passwds  Make    Stamp   Stamp   Stamp
Label            Reqd     XFCBs   Create  Access  Update
--------------   -------  ------  ------  ------  ------
A:Label    .       off      off     off     off     off
Password = SECRET
0A>SET [PROTECT = ON]

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

Directory        Passwds  Make    Stamp   Stamp   Stamp
Label            Reqd     XFCBs   Create  Access  Update
--------------   -------  -----   ------  ------  ------
A:Label    .       on      off     off       off    off
0A>
```

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

The password protection option must be turned on before assigning passwords to individual files or commands. The only exception to this is the directory label password itself. It is the super-password controlling password protection; it is always required when changing the label functions. If no password is assigned to the directory label, or if the password is just a carriage return, then simply press the carriage return key after the Password ? prompt. In the above example, because the password is secret, you type the word secret after the prompt. Note that this password is not echoed on the console. If you make a mistake, you can use a CTRL-X to erase the line and start over, or a CTRL-H to erase the previous character. (See Section 1.4 for an explanation of line-editing control characters).

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

4.20.3 Assigning Passwords to Files

```
0A>SET MYFILE.TXT [PASSWORD = mypsword]
```

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

Note: you should record or remember the passwords you assign to your files. If you do not remember the passwords you assign to your files, you cannot access those files unless password protection is turned off for the whole drive. If you assign a password to the directory label and then forget the password, you will not be able to turn off password protection for the drive.

```
0A>SET MYFILE.TXT [PROTECT = READ]
0A>SET MYFILE.TXT [PROTECT = WRITE]
0A>SET MYFILE.TXT [PROTECT = DELETE]
0A>SET MYFILE.TXT [PROTECT = NONE]
```

When a password is assigned to a file, there are three possible levels of protection afforded to the file by its password. The protection modes are READ, WRITE, DELETE, and NONE (no protection).

When the PROTECT mode is set to READ, you need the password even to read from the file. When the PROTECT mode is set to WRITE, you do not need a password to read from the file, but you do need a password to write to or make any changes to the file.

If the PROTECT mode is set to DELETE, then you do not need the password to read from or make changes to the file, but you do need a password to erase or rename the file.

Finally, when the PROTECT mode is set to NONE, SET erases the password. Files without a password or with a password of blanks always have a protection mode of NONE. Files assigned a password for the first time default to the protection mode of READ.

Table 4-6. Concurrent CP/M-86 Password Protection Modes

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

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

```
0B>SET *.TEX [PASSWORD = SECRET, PROTECT = WRITE]

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

0B>
```

SET displays the password only when a new password is assigned. If you attempt to change the protection mode of a password protected file and do not include the password in the file reference, SET prompts for the password. An advantage to this is that when Concurrent CP/M-86 prompts for the password, it is not echoed to the console and so can not be read by other people watching the screen.

4.20.4 The Default Password

Concurrent CP/M-86 supports an additional password facility called the default password. You can set a default password that is automatically tried whenever a password is required for a file or directory label operation. The default password only applies to the console at which it is set.

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

```
0A>SET [DEFAULT = password]
```

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

4.20.5 Time Stamping of Files

Concurrent CP/M-86 can keep a record of the time and date of the last access and update of a file. Alternatively, the time of creation and last update can be recorded. To enable time stamping, you must turn on the time stamping option for the drive containing the files you wish to time stamp. If the files to be stamped already exist on the drive, you must also individually turn time stamping on for each file.

Concurrent CP/M-86 supports three kinds of date/time stamps.

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

Although there are three kinds of date/time stamps, there can be at most two date/time stamps associated with a given file at one time. The two date/time stamps are recorded in two special fields in the XFCB that can be set up for a file. Once a file has been assigned these stamps, the time stamps can be displayed using the SDIR command (see Section 4.19).

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

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

4.20.6 Time Stamping of New Files

When time stamping is turned on for a drive it automatically applies to all new files subsequently created or copied to the drive.

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

```
        0A>SET [CREATE = ON]
```

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

```
0B>SDIR

Directory For Drive B:

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

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

```
        0A>SET [ACCESS = ON]
```

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

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

```
0B>SDIR

Directory For Drive B:

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

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

```
        0A>SET [UPDATE = ON]
```

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

If you are displaying both update and create date/time stamps, you notice that editing a file changes both the update and create time stamps. This is because the Concurrent CP/M-86 editor, ED (see Section 5) does not update the original file, but renames it with the filetype BAK and creates a new version with the same name as the original.

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

```
0B>SDIR

Directory For Drive B:

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

4.20.7 Time Stamping of Existing Files

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

```
0A>SET MYFILE.TXT [TIME]
```

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

The following three examples show time stamping options.

```
0A>SET [CREATE = ON]
0A>SET [ACCESS = ON]
0A>SET [UPDATE = ON]
```

Each option turns on a special drive option that automatically provides time stamp fields for all new files created or copied to the drive. This option is called the MAKE XFCB option. The MAKE XFCB option automatically makes XFCBs every time a file is created.

It can be independently controlled as shown in the following example.

```
0A>SET [MAKE = OFF]
```

If only a particular group of existing files needs time stamps, you might want to turn the MAKE XFCB option off to prevent the unnecessary creation of Extended File Control Blocks (XFCBs) for other files. The MAKE XFCB option must be ON if CREATE date/time stamps are desired because it assures that a time stamp field will be available at the time a new file is created.

4.20.8 Setting File and Diskette Attributes

In Concurrent CP/M-86, drives and files have a number of special attributes that can be set with the SET command. The attributes fall into two categories: access attributes and identification attributes.

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

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

4.20.9 The Read-Only Attribute

The Read-Only attribute has two modes: R/O (Read-Only) and R/W (Read-Write).

```
0A>SET B:[RO]
0A>SET B:[RW]
```

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

```
0A>SET MYFILE.TXT [RO]
0A>SET MYFILE.TXT [RW]
```

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

4.20.10 The System Attribute

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

```
0A>SET MYFILE.TXT [SYS]
0A>SET MYFILE.TXT [DIR]
```

The SYS option of the SET command sets the file specified by filespec to the type of System. The DIR option sets the file to the type Directory. There are certain drives and user areas from which files can be accessed. When a command file has the SYS attribute and is in user 0 of the system drive, it can be accessed from any location in the system. When a file has the DIR attribute, it can be accessed only if it is in the default user area and default drive (shown in the Concurrent CP/M-86 system prompt). Additionally, system files are not normally displayed by the DIR command.

4.20.11 The Archive Attribute

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

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

```
0A>SET MYFILE.TXT [ARCHIVE = ON]
0A>SET MYFILE.TXT [ARCHIVE = OFF]
```

4.20.12 The User-Definable Attributes

Four user-definable file attributes, F1, F2, F3 and F4, are supported to allow additional identification or classification of certain files. These attributes are not used at all by Concurrent CP/M-86, and exist solely for the use of special applications or your own purposes. They are displayed by STAT and SDIR (see Sections 4.22 and 4.19) and can be set with the following SET commands:

```
        0A>SET MYFILE.TXT [F1 = ON]
        0A>SET MYFILE.TXT [F1 = OFF]
```

In the example above, you can substitute F2, F3, or F4 for F1 to set or reset the file attributes F1, F2, F3, and F4.

4.20.13 Naming Diskettes

When dealing with many diskettes, it is often useful to name or number each diskette. Concurrent CP/M-86 provides a facility for this by creating a directory label for each diskette. The directory label can be assigned an eight character name and three character type similar to a filename and filetype.

```
        0A>SET [NAME=lablname.typ]
```

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

4.20.14 The SET Help Option

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

```
        0A>SET [HELP]
```

4.20.15 Additional Examples

Some examples of possible SET commands follow.

```
        0A>SET *.CMD [SYS,RO,PASS=123,PROT=READ]
        0A>
```

The above setting affords the most protection and the most difficulty for users. Because the password protection mode has been set to READ, you cannot even read one of the CMD files without entering the password 123, unless the default password has been set to 123. Even if the correct password is entered, you still cannot write to the file because the file's Read-Only (R/O) attribute is set ON. Finally, these files will not be displayed by DIR unless the [SYS] option is used.

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

```
0A>SET *.CMD [RW,PROT=NONE,DIR]
Password ?123 <cr>
```

After executing the above command, there is no password protection, the files of type CMD can be read from or written to, and will be listed in all DIR displays. However, they can not be accessed from other user numbers or drives.

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

```
0A>SET *.CMD [RO,SYS]
A:ASM86    CMD  set to system (SYS), read only (RO)
A:DSKMAINT CMD  set to system (SYS), read only (RO)
            .   .   .
A:FUNCTION CMD  set to system (SYS), read only (RO)
A:DDT86    CMD  set to system (SYS), read only (RO)
A:SDIR     CMD  set to system (SYS), read only (RO)
A:SHOW     CMD  set to system (SYS), read only (RO)
A:SET      CMD  set to system (SYS), read only (RO)
```

4.21 The SHOW Command

Syntax:

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

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

The SHOW utility provides the same information as the passive STAT functions (Section 4.22), with the addition of the SHOW LABEL option. SHOW by itself displays the drive, the Read-Only or Read-Write mode for that drive, and the remaining space in kilobytes for all logged in drives in the system. The SHOW SPACE display is the same as the SHOW display. SHOW HELP displays a list of the SHOW options. SHOW with the optional drive specifier displays the SHOW information for the specified drive only, as shown in the following example.

```
0A>SHOW B:

B: RW, Space:     9,488k
```

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

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

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

```
0A>SHOW USERS
```

```
     Active User :  1
     Active Files:  0  2  3  4
```

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

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

You create a directory label when you use a SET command to turn on password protection or time stamping for a drive. Section 4.20 explains the SET commands in detail. The following example illustrates the SHOW LABEL display:

```
Label for drive B:

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

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

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

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

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

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

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

4.22 The STAT (Status) Command

Syntax:

```
STAT d:=R/O
STAT {filespec {$R/O|R/W|SYS|DIR|SIZE}}
STAT {d:}DSK: | USR:
```

The various forms of the STAT utility command give you information about the diskette drives and files associated with your IBM Personal Computer. STAT also lets you change the attributes of files and drives.

The notation R/W tells you the drive is in a Read-Write state, so that data can be both read from and written to the diskette.

The notation R/O tells you the drive is in a Read-Only state, so that data can only be read from, but not written to, the diskette.

Drives are in a Read-Write state by default, and become Read-Only when you set the drive to R/O or when you change a diskette and forget to type a DSKRESET.

Note that the STAT options following filespec can be preceded by a square bracket [, no delimiter or a dollar sign $ as shown above. Note also that the slash / can be omitted from R/O and R/W.

4.22.1 Set a Drive to Read-Only Mode

Syntax:

```
STAT d:= R/O
```

You can use this form of the STAT command to set the drive to Read-Only mode. Use the DSKRESET command to reset a drive to Read-Write.

Example:

```
STAT B:= R/O
```

The command line shown above sets drive B to Read-Only mode.

4.22.2 Free Space on Diskette

Syntax:

```
STAT {d:}
```

STAT with no command tail reports the amount of free storage space that is available on all on-line diskettes. This form of the STAT command reports free space for only those diskettes that have been accessed since Concurrent CP/M-86 was last started. You can find the amount of free space on a particular diskette by including the drive specification in the command tail.

If the drive specifier names a drive that is not on-line, Concurrent CP/M-86 places the drive in an on-line status.

This form of the STAT command displays information on your screen in the following form,

```
d: RW, Free Space: nnK
```

where d is the drive specifier, and n is the number of kilobytes of storage remaining on the diskette in the drive specified by d.

Examples:

```
0A>STAT
```

Suppose you have two diskette drives containing active diskettes. Suppose also that drive A has 16K (16,384) bytes of free space, while drive B has 32K (32,728) bytes of free space. Assume that drive A is marked R/W, and drive B is marked R/O. The STAT command displays the following messages on your screen:

```
A: RW, Free Space: 16K
B: RO, Free Space: 32K
0A>STAT B:
```

Suppose drive B is set to Read-Only and has 98 Kilobytes of storage free for program and data storage. The following message is displayed on your screen:

```
B: RO, Free Space: 98K
```

4.22.3 Files--Display Space Used and Access Mode

Syntax:

```
STAT filespec {$SIZE}
```

This form of the STAT command displays the amount of space in kilobytes used by the specified file. It also displays the Access mode of the file. STAT accepts wildcards in the filename and filetype part of the command tail. When you include a wildcard in your file specification, the STAT command displays a list of qualifying files from the default or specified drive, with their file characteristics, in alphabetical order.

Concurrent CP/M-86 supports four file Access modes:

- R/O The file has the Read-Only attribute that allows data to come from the file, but the file cannot be altered.
- R/W The file has the Read-Write attribute that allows data to move either to or from the file.
- SYS The file has the system attribute. System files do not appear in DIR (directory) displays. Use the STAT command to display all files including those with the System attribute. The STAT command shows System files in parentheses.
- DIR The file has the directory attribute and appears in DIR (directory) displays.

A file has either the R/O or R/W attribute, and either the SYS or DIR attribute. By default, and unless changed by the STAT command, a file has the R/W and DIR attributes.

The SIZE option in a STAT command produces a list of file characteristics under five headings.

The first column displays the number of records used by the file, where each record is 128 bytes in length. This value is listed on your screen under the column marked Recs.

The second column displays the number of kilobytes used by the file, where each kilobyte contains 1,024 bytes. This value is listed under Bytes.

The third column displays the number of directory entries used by the file. This value appears under the FCBs column. FCB (File Control Block) is another name for a directory entry.

The Access Modes are displayed under the Attributes column.

The file specification, consisting of the drive specification, filename, and filetype of the file appears under Name on your screen.

If the drive specifier is included, and the corresponding drive is not active, Concurrent CP/M-86 places the drive in an active status.

Use $SIZE to tell STAT to compute the virtual file size of each file. The virtual and real file size are identical for sequential files, but can differ for files written in random mode. When you use $SIZE, the additional column, marked Size, is displayed on the screen. The value in this column represents the number of filled and unfilled records allotted to the file.

When you enter the command STAT *.*, STAT performs a directory verification to ensure that two files do not share the same disk space allocation. This means that the indicated file shares a portion of the diskette with another file in the directory. If STAT finds a duplicate space allocation, it displays the following message:

```
Bad Directory on d:

Space Allocation Conflict:

d:filename.typ
```

STAT prints the name of the file containing doubly-allocated space. More than one file can be listed. The recommended solution is to erase the listed files.

STAT does a complete directory verification whenever a wildcard character appears in the command tail.

Examples:

```
0A>STAT MY*.*
```

This command tells STAT to display the characteristics of all files that begin with the letters MY, with any filetype at all. Assume that the following three files satisfy the file specification. The screen could display the following:

```
Recs  Bytes FCBs Attributes      Name

 16     2K    1 Dir RW        B:MYPROG.A86

  8     1K    1 Dir RO        B:MYTEST.DAT

 32    18K    2 Sys RO        B:MYTRAN.CMD

Total:  21K    4

B: RW, Free Space: 90K
```

```
0A>STAT MY*.* $SIZE
```

This command causes the same action as the previous command, but includes the Size column in the display. Assume that MYTEST.DAT was written using random access from record number 8 through 15, leaving the first 8 records empty. The virtual file size is 16 records, although the file consumes only eight records. The screen appears as follows:

```
Size Recs  Bytes FCBs Attributes      Name

 16   16     2K    1 Dir RW        B:MYPROG.A86

 16    8     1K    1 Dir RO        B:MYTEST.DAT

 32   32    18K    2 Sys RO        B:MYTRAN.CMD

Total:      21K    4

B: RW, Free Space: 90K
```

4.22.4 Set File Access Modes (Attributes)

Syntax:

```
STAT filespec $R/O |$R/W |$SYS |$DIR
```

This form of the STAT command lets you set the Access mode for one or more files. The four Access modes, described above, are:

```
R/O    R/W    SYS    DIR
```

If the drive named in the file specification corresponds to an inactive drive, Concurrent CP/M-86 first places the drive in an on-line state.

A file can have either the R/O or R/W Access mode, but not both. Similarly, a file can have either the SYS or DIR Access mode, but not both.

Examples:

```
0A>STAT LETTER.TXT $R/O
```

This command sets the Access mode for the file LETTER.TXT on the default drive to R/O. The following message appears on your screen if the file is present:

```
LETTER.TXT set to R/O
0B>STAT A:*.CMD $SYS
```

This command sets the Access mode for all files on drive A, with filetype CMD, to SYS. Given that the three command files PIP, ED, and ASM86 are present on drive A, the following message appears on your screen:

```
PIP.CMD set to SYS

ED.CMD set to SYS

ASM86.CMD set to SYS
```

4.22.5 Display Diskette Status

<u>Syntax:</u>

```
STAT {d:}DSK:
```

This form of the STAT command displays internal information about your diskette system for all on-line diskettes.

If a drive is specified, it is placed in an on-line status.

The information provided by this command is useful for more advanced programming, and is not necessary for everday use of Concurrent CP/M-86.

The third entry in the display, Kilobyte Drive Capacity, shows the total amount of storage in Kilobytes available on the specified diskette. After logging in a new diskette, using the DSKRESET command, you can use this form of the STAT command to determine whether the diskette is single-sided or double-sided. Single-sided diskettes have a total of 156 kilobytes of disk storage. Double-sided diskettes have a total of 316 kilobytes of storage.

<u>Examples:</u>

```
0A>STAT DSK:
```

This STAT command displays information about drive A in the following form. STAT supplies numbers for n.

```
    A: Drive Characteristics
 nnnn: 128 Byte Record Capacity
 nnnn: Kilobyte Drive  Capacity
 nnnn: 32  Byte Directory Entries
 nnnn: Checked  Directory Entries
 nnnn: 128 Byte Records/Directory Entry
 nnnn: 128 Byte Records/Block
 nnnn: 128 Byte Records/Track
 nnnn: Reserved Tracks
0A>STAT B:DSK:
```

This command produces the information shown in the previous example for drive B.

4.22.6 Display User Numbers With Active Files

Syntax:

```
0A>STAT {d:}USR:
```

Concurrent CP/M-86 assigns the current user number to files created under it. Except when using the [Gn] option with a PIP command, or accessing a file in user 0 that has been set to SYS with the STAT command, you can only access that file from the user number under which it was created. This form of the STAT command displays the current user number and the user numbers containing files on the diskette in the default or specified drive.

Examples:

```
0A>STAT USR:

A: Active User:  0
A: Active Files: 0 15
```

The STAT USR: command with no drive specification displays information for the default drive. This command displays the current user number and the user numbers that contain files in the form shown above. The display begins with the default or specified drive. STAT displays the current user number following Active User. In this case it is user 0. The user numbers containing files are displayed following Active Files. In the example above user 0 and user 15 contain files.

4.23 The SUBMIT (Batch Processing) Command

Syntax:

```
SUBMIT filespec { parameters... }
```

Normally, you enter commands one line at a time. If you must enter the same sequence of commands several times, you will find it easier to batch the commands together using the SUBMIT utility. To do this, create a file and list 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 filespec and prepares it for interpretation by Concurrent CP/M-86.

The file of type SUB can contain any valid Concurrent CP/M-86 commands. A command line cannot exceed 125 characters. The file of type SUB can contain a maximum of 128 command lines. The SUBMIT buffer allows up to 2048 characters in the input file.

If you want, you can include SUBMIT parameters within the SUB file that are filled in by values that you include in the command tail. SUBMIT parameters take the form of a dollar sign, $, followed by a number in the range 0 through 9:

```
$0  $1  $2  $3  $4  $5  $6  $7  $8  $9
```

You can put these parameters anywhere in the command lines in your file of type SUB.

The SUBMIT utility reads the command line following SUBMIT filespec and substitutes the items you type in the command tail for the parameters that you included in the file of type SUB. When the substitutions are complete, SUBMIT sends the file to Concurrent CP/M-86 line-by-line as if you were typing each command.

Each item in the command tail is a sequence of alphabetic, numeric, or special characters. The items are separated by one or more blanks.

The first word in the command tail takes the place of $1 in the SUB file, the second word replaces $2, and so forth, through the last item in the command tail. The filename of the SUB file replaces any $0 parameters in the SUB file.

SUBMIT creates a file named $$$.SUB that contains the command lines resulting from the substitutions.

If you type fewer items in the command tail than parameters in the SUB file, remaining parameters are not included in the temporary file.

If you type more items in the command tail than parameters in the SUB file, the items remaining in the command tail are ignored.

Batch command processing stops after reading the last line of the $$$.SUB file. CTRL-C or CTRL-BREAK stops the SUBMIT process. You can also stop batch processing before reaching the end of the SUB file by pressing any key after Concurrent CP/M-86 issues the command input prompt, 0A>.

The file $$$.SUB is automatically removed when Concurrent CP/M-86 stops reading the batched commands.

SUB files cannot contain nested SUBMIT commands. However, the last command in a SUB file can be a SUBMIT command that chains to another SUB file.

To include an actual dollar sign in your file of type SUB, type two dollar signs, $$. The SUBMIT utility replaces then with a single dollar sign when it substitutes a command tail item for a $ parameter in the SUB file.

Passwords

SUBMIT does not support passwords. If your files require passwords, be sure they all have the same one, and set the default password to this password.

Examples:

```
0A>SUBMIT SUBFILE
```

Assume the file SUBFILE.SUB is on the diskette in drive A, and that it contains the lines shown below.

```
DIR *.COM

ASM86 X $$SB

PIP LST:= X.PRN[T8D80]
```

The SUBMIT command shown above sends the sequence of commands contained in SUBFILE.SUB to Concurrent CP/M-86 for processing. Concurrent CP/M-86 first performs the DIR command and then assembles X.A86. When ASM86 finishes, the PIP command line is executed.

```
0A>SUBMIT B:ASMCOM X  8 D80  SZ  <--these command tail
                                     items are assigned
                   $1 $2  $3  $4  <--to these SUB file $n
                                     parameters.
```

Assume that ASMCOM.SUB is present on drive B and that it contains the commands:

```
ERA $1.BAK

ASM86 $1 $$4

PIP LST:= $1.PRN[T$2 $3 $5]
```

The SUBMIT utility reads this file and substitutes the items in the command tail for the parameters in the SUB file as follows:

```
ERA X.BAK

ASM86 X $SZ

PIP LST:= X.PRN[T8 D80]
```

These commands are executed from top to bottom by Concurrent CP/M-86.

4.24 The SYSDISK (Display and Set System Diskette) Command

Syntax:

```
SYSDISK
SYSDISK D:
```

The SYSDISK command tells you which drive is the current system drive and lets you specify a new system drive. Valid drive specifications are in the range A, B, C, D, and M.

SYSDISK is especially useful within SUBMIT programs. You can create a start-up file with a SUBMIT command that sets your system diskette to MDISK, and then transfers all your system files there, freeing your old system drive for other work.

There are three error messages you can receive from SYSDISK:

```
Invalid drive code, use following syntax SYSDISK A:

Unable to set new system disk because of open files

Specified device does not exist
```

Examples:

Give the SYSDISK command to return the current system diskette drive:

```
0A>SYSDISK

System disk is A:
```

Give the command SYSDISK with a drive letter A, B, C, D, or M:

```
0A>SYSDISK M:

System disk is M:
```

4.25 The TOD (Display and Set Time of Day) Command

<u>Syntax:</u>

```
TOD {time-specification | P}
```

The TOD utility lets you examine and set the time of day.

A date is represented as a month value in the range 1 to 12, a day value in the range 1 to 31, depending upon the month, and a two digit year value relative to 1900.

Time is represented as a 24 hour clock, with hour values from 00 to 11 for the morning, and 12 to 23 for the afternoon.

Use the command,

```
TOD
```

to obtain the current date and time in the format:

```
weekday  month/day/year  hour:minute:second
```

For example, the screen might appear as,

```
Fri 12/10/82  14:15:23
```

in response to the TOD command.

Use the command form,

```
TOD time-specification
```

to set the date and time, where the time-specification takes the form:

```
month/day/year hour:minute:second
```

A command line in this form is:

```
TOD 02/09/83 10:30:00
```

To let you accurately set the time, the TOD utility writes the message:

```
Press any key to set time
```

When the time that you give in the command tail occurs, press any key. TOD begins timing from that instant, and responds with a display in the form:

```
Wed 02/09/83 10:30:00
```

Use the command form,

```
TOD P
```

to print the date and time continuously on the screen. This display appears in the status line also. However, you can assign the screen to another logical device on which you might want to display the date and time.

You can stop the continuous display by pressing any key.

You need not set the time of day for proper operation of Concurrent CP/M-86.

Examples:

```
0A>TOD
```

This command writes the current date and time on the screen.

```
0A>TOD 12/31/82 23:59:59
```

This command sets the current date and time to the last second of 1982.

4.26 The TYPE (Display File) Command

Syntax:

```
TYPE filespec
```

The TYPE command displays the contents of a character file on your screen.

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

Press any key on your keyboard to discontinue the TYPE command.

Make sure the file specification identifies a file containing character data.

If the file named by filespec is not present on an on-line diskette, TYPE displays the following message on your screen:

```
NO FILE
```

To list the file at the printer as well as on the screen, type a CTRL-P before entering the TYPE command line. To stop echoing keyboard input at the printer, type a second CTRL-P.

TYPE returns the following error message if you give it an ambiguous filespec; TYPE can only type one file at a time.

```
No wildcards allowed
```

Examples:

```
0A>TYPE MYPROG.A86
```

This command displays the contents of the file MYPROG.A86 on your screen.

```
0A>TYPE B:THISFILE
```

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

4.27 The USER (Display and Set User Number) Command

Syntax:

```
USER { number }
```

The USER command displays and changes the current user number. The diskette directory can be divided into distinct areas according to a user number.

The default user number is 0. When Concurrent CP/M-86 starts, it assumes that 0 is the current user number. Your IBM Personal Computer displays the current user number in the prompt as the number immediately before the drive letter:

```
0A>
```

This number can be in the range zero through fifteen. Any files you create under this user number are not accessible under any other user number except through the PIP command. (See the G parameter of the PIP utility.)

Use the command,

```
USER
```

to display the current user number.

Use the command,

```
USER number
```

where number is a number in the range 0 through 15, to change the current user number.

Give the command,

```
SHOW USR:
```

to obtain a list of user numbers that have files associated with them.

Examples:

```
0A>USER
```

This command displays the current user number.

```
0A>USER 3
```

This command changes the current user number to 3.

4.28 The VCMODE (Set Virtual Console Background Mode) Command

Syntax:

```
VCMODE
VCMODE dynamic
VCMODE buffered
VCMODE size = n
```

The VCMODE command lets you specify background operating modes for the four virtual consoles.

When a Concurrent CP/M-86 virtual console is switched-out, or in the background, it operates in one of two possible modes: the Dynamic mode or the Buffered mode.

If a console is in Dynamic mode and you switch it into the background, data normally output to the screen fills a space in memory reserved for such data. If there is enough data output to more than fill this area, the oldest data is lost as the new data is written in, just as when new lines appear on the bottom of the screen as old ones disappear from the top. In this case, when you switch back to this console some data may be lost.

If a console is in Buffered mode and switched-out, data output from a running program goes into a buffer file on diskette rather than to the screen storage area in memory. Concurrent CP/M-86 looks for the highest-lettered drive in the system and writes these buffer files onto this diskette. If you have implemented the MDISK, the system will always write buffer files there. When you switch in this console, the system fetches the data stored in the buffer file on diskette and writes it onto the screen. These files disappear after their data appears on the screen.

You can use the control character CTRL-S to stop data scrolling past your screen and CTRL-Q to restart the data.

You can specify the maximum size of the diskette buffer file with the VCMODE command. If a buffer file fills up, Concurrent CP/M-86 suspends that console's program operation; no data is lost. Operation begins when you switch this console in again. You cannot specify a buffer size for a console set to Dynamic mode. If you try to, Concurrent CP/M-86 merely reports that the console is set to Dynamic Mode, and nothing else happens.

Try out these examples and you will quickly become familiar with how Concurrent CP/M-86 manages multiple virtual consoles.

Examples:

If you ask for help, or type anything other than a VCMODE command, VCMODE explains itself briefly:

```
0A>VCMODE HELP

                VCMODE EXAMPLES

vcmode                  (show background mode)
vcmode dynamic          (sets background mode)
vcmode buffered         (sets background mode)
vcmode size = 5         (sets buffered mode max file size in)
                        (kilobytes, legal range is 1 to 8191)
vcmode help             (prints this message)
```

Typing VCMODE by itself returns a line telling you what the background mode for your current console is:

```
0A>vcmode

Background Mode For Virtual Console    0 is Dynamic
```

Entering VCMODE DYNAMIC returns a line confirming the command:

```
0A>vcmode dynamic

Background Mode For Virtual Console    0 set to Dynamic
```

Entering VCMODE BUFFERED returns a line confirming the command and tells you how large the maximum file size is:

```
0A>vcmode buffered

Background Mode For Virtual Console    0 set to Buffered
Maximum file size =    12K
```

If you try to specify the maximum file size when the current console is in Dynamic mode, you get:

```
0A>vcmode size=5

Background Mode For Virtual Console    0 set to Dynamic
```

Set it to Buffered Mode first:

```
0A>vcmode buffered

Background Mode For Virtual Console    0 set to Buffered
Maximum file size =    12K
```

Then, specify the maximum file size:

```
0A>vcmode size=5

Background Mode For Virtual Console   0 set to Buffered
Maximum file size =     5K
```

End of Section 4

Section 5
The Concurrent CP/M-86 Editor

5.1 Introduction to ED

To do almost anything with a computer you need some way to enter data, some way to give the computer the information you want it to process. The programs most commonly used for this task are called editors. They transfer your keystrokes at the keyboard to a diskette file. Concurrent CP/M-86's editor is named ED. Using ED, you can easily create and alter Concurrent CP/M-86 text files.

The correct command format for invoking the Concurrent CP/M-86 editor is given in Section 5.2, Starting ED. After starting ED, you issue commands that transfer text from a disk file to memory for editing. Section 5.3, ED Operation, details this operation and describes the basic text transfer commands that allow you to easily enter and exit the editor.

Section 5.4, Basic Editing Commands, details the commands that edit a file. Section 5.5, 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 Section 5.6, Advanced ED Commands.

During an editing session, ED can return two types of error messages. Section 5.7, ED Error Messages, lists these messages and provides examples that indicate how to recover from common editing error conditions.

5.2 Starting ED

Syntax:

```
        ED input-filespec output-filespec
```

To start ED, enter its name after the Concurrent CP/M-86 prompt. The command ED must be followed by a file specification, one that contains no wildcard characters, such as:

```
        0A>ED MYFILE.TEX
```

The file specification, MYFILE.TEX in the above example, specifies a file to be edited or created. The file specification can be preceded by a drive specification, but a drive specification is unnecessary if the file to be edited is on your default drive. Optionally, the file specification can be followed by a drive specification, as shown in the following example.

```
0A>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 shown in the following example.

```
0A>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 shown below.

```
0A>ED MYFILE.TEX
    : *
```

If no previous version of the file exists on the current diskette, ED automatically creates a new file and displays the following message:

```
NEW FILE
    : *
```

Note: before starting an editing session, use the STAT command to check the amount of free space on your diskette. Make sure that the unused portion of your diskette is at least as large as the file you are editing--larger if you plan to add characters to the file. When ED finds a diskette or directory full, ED has only limited recovery mechanisms. These are explained in Section 5.7, ED Error Messages.

5.3 ED Operation

With ED, you change portions of a file that pass through a memory buffer. When you start ED with one of the commands shown above, 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. The following figure illustrates the relationship between the source file, the temporary work file and the new 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 diskette if the output file is written to another diskette.

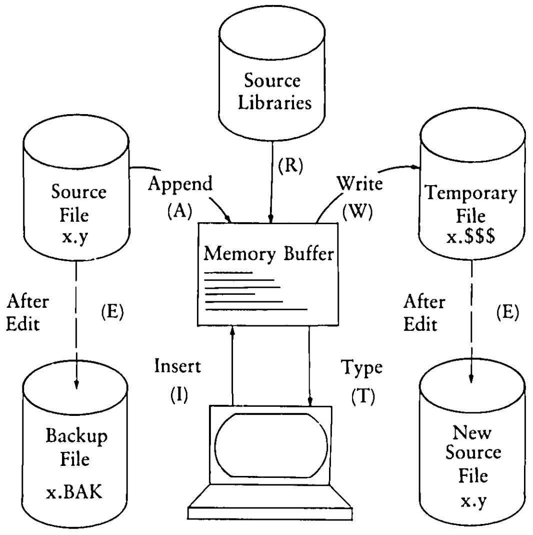


Figure 5-1. Overall ED Operation

In the preceding figure, 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 5-1. 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.

Table 5-1. (continued)

Command	Result
E	End the edit. Copy all buffered text to the temporary file, and copy all unprocessed source lines to the temporary file. Rename files.

5.3.1 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 turned 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 forms of the V command are:

```
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 shown below.

```
46652/48253
```

The A (Append) Command

The A command appends (copies) lines from an existing source file into the memory buffer. The form of the A command is,

```
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, than the integer 65535 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.

5.3.2 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 form of the W command is,

```
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 form of the E command is,

```
E
```

followed by a carriage return.

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.

5.4 Basic Editing Commands

The text transfer commands discussed above 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 together 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 carriage return. 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:

- If the numeric argument is omitted, ED assumes an argument of 1.
- Use a negative number if the command is to be executed backwards through the memory buffer. (The B command is an exception.)
- If you enter a pound sign, #, in place of a number, ED uses the value 65535 as the argument. A pound sign argument can be preceded by a minus sign to cause the command to execute backwards through the memory buffer (-#).
- ED accepts 0 as a numeric argument only in certain commands. In some cases, 0 causes the command to be executed approximately half the possible number of times, while in other cases it prevents the movement of the CP.

The following table alphabetically summarizes the basic editing commands and their valid arguments.

Table 5-2. 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 backward (-nC) through the memory buffer.
nD, -nD	Delete n characters before (-nD) or after (nD) the CP.
I	Enter insert mode.
Istring CTRL-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 backward (-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 Section 5.6, Combining ED Commands, 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.

5.4.1 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 forms of the B command are:

```
        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 forms of the C command are,

```
        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 forms of the L command are,

```
        nL, -nL
```

where n is the number of lines the CP is to be moved. A positive number moves the CP towards the end of the buffer. A negative number moves the CP 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 forms of the n command are,

```
n, -n
```

where n is the number of lines the CP is to be moved. In response to this command, ED moves 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 carriage return without a preceding command, ED assumes a n command of 1 and moves the CP down to the next line and prints it, as shown below.

```
 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.

5.4.2 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 forms of the T command are,

```
nT, -nT
```

where n specifies the number of lines to be displayed. If a negative number is entered, ED displays n lines before the CP. A positive number displays n lines 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 memory buffer example below.

```
"I find ec^stasy in living -
```

Whenever ED is displaying text with the T command, you can enter a CTRL-S to stop the display, then press any key when you are ready to continue scrolling. Enter a CTRL-C to abort long type-outs.

5.4.3 Deleting Characters

The D (Delete) Command

The D command deletes a specified number of characters and has 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 shown below,

```
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 the <cr><lf> between two lines to join them together. 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.

5.4.4 Inserting Characters into the Memory Buffer

The I (Insert) Command

To insert characters into the memory buffer from the screen, use the I command. If you enter the command in upper-case, ED automatically converts the string to upper-case. 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.

```
0A>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 insert or command mode, you can use Concurrent CP/M-86 line-editing control characters to edit your input. Note however, that you cannot use CTRL-E in insert mode. The table below lists these control characters.

Table 5-3. Concurrent CP/M-86 Line Editing Controls

Command	Result
CTRL-E	Return carriage for long lines without transmitting command line to the buffer.
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.
Backspace	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 upper-case of numbers to special characters. ED provides two ways to translate your alphabetic input to upper-case without affecting numbers. The first is to enter the insert command letter in upper-case: I. All alphabetics entered during the course of the capitalized command, either in insert mode or as a string, are translated to upper-case. (If you enter the insert command letter in lower-case, all alphabetics are inserted as typed). The second method is to enter a U command before inserting text. Upper-case 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 Concurrent CP/M-86'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 shown below:

```
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 shown below:

```
In 1870, ^Emily Dickinson said,<cr><lf>
```

5.4.5 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 upper-case, ED automatically converts the string to upper-case. 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 shown below:

```
The poet^ said,<cr><lf>
```

If upper-case 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.

5.5 Combining ED Commands

It saves keystrokes and editing time to combine 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 the carriage return key. Use Concurrent CP/M-86's line-editing controls to manipulate ED command strings.

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

- The combined-command line must not exceed Concurrent CP/M-86's 128 character maximum.
- If the combined-command line contains a character string, the line must not exceed 100 characters.
- Commands to terminate an editing session must not appear in a combined-command line.
- Commands, such as the I, S, J, X, and R commands, that require character strings or filespecs 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.

5.5.1 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: *
```

5.5.2 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 example below, 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.

```
    3: *T
sense of living
    3: *0LT
    3:  the mere sense of living
    3: *
```

The command 0TT displays the entire line without moving the CP.

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: *
```

5.5.3 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 below 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 shown below.

```
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 shown below. 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: *
```

5.6 Advanced ED Commands

The basic editing commands discussed above allow you to use ED for all your editing. The following ED commands, however, enhance ED's usefulness.

5.6.1 Moving the CP and Displaying Text

<u>The P (Page) Command</u>

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. To do this, use 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 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 backwards 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 form for the line number command is,

```
n:
```

where n is the number of the destination line. This command places the CP at the beginning of the specified line. 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 only with three ED commands: the T (type) command, the L (line) command, and the K (kill) 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, as shown below.

```
1: *2::4T
2:  "I find ecstasy in living -
3:  the mere sense of living
4:  is joy enough."
2: *
```

5.6.2 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 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. Its form is,

```
nFstring
```

where n is the occurrence of the string to be found. Any number you enter must be positive because ED can only search from the CP to the bottom of the buffer. If you enter no number, ED finds the next occurrence of the string in the file. In the following example, the second occurrence of the word living is found.

```
1: *2fliving
3: *
```

The character pointer moves to the beginning of the third line where the second occurrence of the word "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 shown below.

```
1: *2fliving^Z0lt
3: *the mere sense of living
```

It makes a difference whether you enter the F command in upper- or lower-case. If you enter F, ED internally translates the argument string to upper-case. If you specify f, ED looks for an exact match. For example, FConcurrent CP/M-86 searches for CONCURRENT CP/M-86 but fConcurrent CP/M-86 searches for Concurrent CP/M-86, and cannot find CONCURRENT CP/M-86.

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 form of the N command is,

```
nNstring
```

where n is the occurrence of the string to be found. If no number is entered, ED looks for the next occurrence of the string in the file. The case of the N command 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 a third delete-to string. This juxtaposes the string between the search and delete-to strings with the insert string. The form of the J command is,

```
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" and 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. (This is shown in the following example.) If an upper-case J command letter is specified, ED looks for upper-case search and delete-to strings and inserts an upper-case insert string.

The J command is especially useful when revising comments in assembly language source code, as shown below.

```
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 your desired phrase 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 the carriage return key 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 form of the M command is,

```
nMcommand string
```

where n is the number of times the command string is to be executed. A negative number is not a valid argument for an M command. If no number is specified, the special character # is assumed, and ED executes the command string until it reaches the end of data in the buffer or the end of 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 carriage return; therefore, an M command must be the last command on the line. Also, all character strings that appear in a macro must be terminated by CTRL-Z or ESC. If a character string ends the combined-command string, it must be terminated by CTRL-Z, then followed by a <cr> 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 a CTRL-C at the keyboard.

The Z (Sleep) Command

Use the Z command to make the editor pause between operations. The pauses give you a chance to review what you have done. The form of the Z command is,

```
nZ
```

where n is the number of seconds to wait before proceeding to the next instruction.

Generally, the Z command has no real effect unless you use it with a macro command. The example below shows you how you can use the Z command to cause a ten second pause each time ED finds the word TEXT in a file.

```
0A>*mfliving^Z0tt10z
```

5.6.3 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
Rfilepsec
```

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: *41
 : *R^ZB#T
1:  "I find ecstasy in living -
2:  the mere sense of living
3:  is joy enough."
4:  Emily Dickinson said,
1: *
```

5.6.4 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 form of the H command is,

```
H
```

followed by a carriage return.

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:

```
0A>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 form of the O command is,

```
O
```

followed by a carriage return. When you enter an O command, ED confirms that you want to abandon your changes by asking:

```
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 form of the Q command is,

```
Q
```

followed by a carriage return.

When you enter a Q command, ED verifies that you want to abandon the changes by asking:

```
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 Concurrent CP/M-86.

Note: you can enter a CTRL-Break or a CTRL-C to return control immediately to Concurrent CP/M-86. This does not give ED a chance to close the source or new files, but it prevents ED from deleting any temporary files.

5.7 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 Concurrent CP/M-86 error message if ED cannot read or write to the specified file. The form of an ED error message is,

```
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 5-4. **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, Q, or O 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 diskette 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 diskette or a B#Xd:buffer.sav command to write the contents of the memory buffer onto another diskette.

```
BREAK "F" AT n
DIRECTORY FULL
```

Use the same commands described in the previous message to require from this file error.

The following table defines the diskette file error messages ED returns when it cannot read or write a file.

Table 5-5. ED Diskette File Error Messages

Message	Meaning
Bdos Err on d: R/O Function NNN File: FILENAME.TYP	Disk d: has Read-Only attribute. This occurs if a different diskette has been 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.
File Not Available	The file specified in the command is open on another virtual console, and ED cannot get access to it.

End of Section 5

Appendix A
Concurrent CP/M-86 Error Messages

Error messages come from several different sources. Concurrent CP/M-86 displays error messages generated by mistakes in command lines. Some errors emerge from individual utilities, such as the errors associated with SYSDISK. Diskette and file errors come from the BDOS module. Other errors come from the XIOS module. You can also intercept errors from applications programs you are running. Table A-1 displays the BDOS error messages. Table A-2 lists XIOS error messages. Table A-3 shows command line errors.

BDOS error messages are displayed in the following format:

```
    Bdos Err on d:  Message
    Function NNN    File: FILENAME.TYP
```

"d" indicates the drive involved; "message" indicates the appropriate error message; NNN indicates the number of the BDOS function involved, and FILENAME.TYP indicates the file involved.

Table A-1. Concurrent CP/M-86 Error Messages

Message	Meaning
File Opened in Read/Only Mode	A process attempted to write to a file opened in Read-Only mode. A file is opened in Read-Only mode if it resides in User 0 and is opened from another user number. This error also occurs when a process attempts to write to a password-protected (write mode) file when an incorrect password was supplied with the open call.
File Currently Open	A process tried to delete, rename, or modify an attribute of a file opened by another process. This error also occurs when a process tries to open a file that is already open, and the new open mode is incompatible with the mode in which the file was opened previously.

Table A-1. (continued)

Message	Meaning
Close Checksum Error	A file cannot be closed properly because the directory Concurrent CP/M-86 finds on the diskette does not match the one logged into memory. This probably means the diskette has not been reset, or a program error has occurred. Give the DSKRESET command. **Note:** If you receive this error and then remove the specified diskette, you might lose data on the diskette and also on any diskette placed in that drive. Give the DSKRESET command after inserting the new diskette. This command initializes the diskette, but data on the diskette previously inserted might be lost.
Password Error	A password is incorrect or missing.
File Already Exists	You tried to create or rename a file when there is already a file of that filename and filetype on the specified drive and user number.
Illegal ? in FCB	You used a wildcard character where wildcard filespecs are not permitted.
Open File Limit Exceeded	You tried to open one more file than the maximum number of open files per process that the system can accommodate.
No Room in System Lock List	The Open File, Make File, and Access Drive function return this error when no room for new entries exists within the system lock list.

Table A-1. (continued)

Message	Meaning
Bad Sector	This error occurs when there is an actual hardware error on the diskette. It can occur as a result of trying to read a diskette of one density in a drive set to a different density. This error also occurs with an improperly inserted floppy diskette.
Select	You selected a nonexistent drive.
R/O	A process tried to write to a file when the drive was set to Read-Only.
File R/O	A process tried to write to a file when that file was set to Read-Only.

Table A-2. XIOS Error Messages

Message	Meaning
Parity Error Parity error from <processname> at <address> **** MACHINE IS HALTED! ****	Some memory operation generated a parity interrupt, not necessarily in this process at this address. There is no recovery procedure; turn power off and on again.

Table A-2. (continued)

Message	Meaning
Unexpected Interrupt Unexpected interrupt from <processname> at <address> Press any key to stop process	Concurrent CP/M-86 has trapped an interrupt it cannot deal with. In many cases such interrupts are generated by applications software calls to the IBM ROS. Digital Research does not support ROS calls in Concurrent CP/M-86.
Overflow Interrupt Overflow interrupt from <processname> at <address> Press any key to stop process	An arithmetic overflow has been generated by a multiply or divide operation. Note that the 8088 assembly language contains an overflow enable instruction; if this instruction is not given, overflow interrupts cannot occur.
Divide Interrupt Divide interrupt from <processname> at <address> Press any key to stop process	An illegal divide operation has occurred. This error can be a divide-by-zero or some other divide error.
Accept,Retry,Ignore,Details? Error Messages	Certain XIOS errors generate error messages that offer four options to you during an error report. The error messages associated with the printer and with some diskette operations return the following type of error report:
Printer error Accept,Retry,Ignore,Details? Disk error Accept,Retry,Ignore,Details?	Each selection causes a different course of action.

Table A-2. (continued)

Message	Meaning
	"A" (Accept) makes the system continue without handling the error in any special way. In some cases the BDOS responds with another error; in other cases an application program might trap the original error on its own. "R" (Retry) repeats the operation that caused the error. The XIOS repeats this operation locally, without involving the rest of Concurrent CP/M-86; if the error occurs again, this information is not passed back to the system. The operation can take place so rapidly that the error message never seems to leave the screen, but merely flickers for a moment. "I" (Ignore) returns you to system level, taking no note of the error. If the error condition still exists, you will encounter the error message again. "D" (Details) shows the specifics of the problem. Your course of action depends on the nature of the error. See the examples below for descriptions of the details error reports.
Printer Errors	
	There are two kinds of printer errors; both use the same reporting mechanism.
Printer error Accept,Retry,Ignore,Details?	
	If you press D for Details, the following message appears. Either or both of the two messages within parentheses can occur:
Printer <number> (out of paper) (not online) A/R/I/D?	
	Inspect the printer indicated by <number> and retry.
Disk Errors	
	There are seven kinds of disk errors reported from the XIOS. All use the same format.

Table A-2. (continued)

Message	Meaning
Disk error	Accept,Retry,Ignore,Details? If you type D this detail list appears:
d: Track nn Sector nn Side n <error report> A/R/I/D?	where d: is the drive number, Track nn is the track number, Sector nn is the sector number, and <error report> is one of these six words or phrases: write read CRC DMA not found write protect no address mark

Table A-3. Concurrent CP/M-86 System Error Messages

Message	Meaning
?Load Error	A physical error occurred while loading a CMD file.
?Not Enough Memory	There is not a large enough memory in which to load a file of type CMD.
?PD Table Full	Number of Process Descriptors specified at GENSYS has been exceeded.
?RSP Command Que Full	Queue full: the queue specified in the command keyword cannot hold any more messages.

Table A-3. (continued)

Message	Meaning
?Bad File Spec	You entered an improperly formatted command keyword.
?Can't Find Command	Concurrent CP/M-86 cannot find your command file.
?CLI Abort	The Command Line Interpreter was stopped, typically by a CTRL-C.
?Read Error	The CLI did not read the command file code correctly.

Table A-4. Concurrent CP/M-86 Utility Error Messages

Message	Meaning
ambiguous operand	DDT-86. An attempt was made to assemble a command with an ambiguous operand. Precede the operand with the prefix "BYTE" or "WORD".
Bad Directory on d: Space Allocation Conflict: User n d:filename.typ	STAT has detected a space allocation conflict in which one data block is assigned to more than one file. One or more filenames might be listed. Each of the files listed contain a data block already allocated to another file on the disk. You can correct the problem by erasing the files listed. After erasing the conflicting file or files, press CTRL-C to regenerate the allocation vector. If you do not, the error might repeat itself.

Table A-4. (continued)

Message	Meaning
BDOS err on d:	Concurrent CP/M-86 replaces d: with the drive specifier of the drive where the error occurred. This message appears when Concurrent CP/M-86 finds no disk in the the drive, when the disk is improperly formatted, when the drive latch is open, or when power to the drive is off. Check for one of these situations and retry.
Cannot close	ASM-86. An output file cannot be closed. This is a fatal error that terminates ASM-86 execution. You should take appropriate action after checking to see if the correct disk is in the drive and that the disk is not write protected. DDT-86. The disk file written by a W command cannot be closed. This is a fatal error that terminates DDT-86 execution. You should take appropriate action after checking to see if the correct disk is in the drive and that the disk is not write protected.
Command name?	If Concurrent CP/M-86 cannot find the command you specified, it returns the command name you entered followed by a question mark. Check that you have typed the command name correctly, or that the command you requested exists as a .CMD file on the default or specified disk.
DESTINATION IS R/O, DELETE (Y/N)?	PIP. The destination file specified in a PIP command already exists and it is Read-Only. If you type Y, the destination file is deleted before the file copy is done.

Table A-4. (continued)

Message	Meaning
Directory full	
	ASM-86. There is not enough directory space for the output files. You should either erase some unnecessary files or get another disk with more directory space and execute ASM-86 again.
Disk full	
	ASM-86. There is not enough disk space for the output files (LST, H86 and SYM). You should either erase some unnecessary files or get another disk with more space and execute ASM-86 again.
Disk read error	
	ASM-86. A source or include file could not be read properly. This is usually the result of an unexpected end of file. Correct the problem in your source file. DDT-86. The disk file specified in an R command could not be read properly. This is usually the result of an unexpected end of file. Correct the problem in your file.
Disk write error	
	DDT-86. A disk write operation could not be successfully performed during a W command, probably due to a full disk. You should either erase some unnecessary files or get another disk with more space and execute ASM-86 again.
Double defined variable	

	ASM-86. An identifier used as the name of a variable is used elsewhere in the program as the name of a variable or label. Example:
	
	`X        DB   5`
	`          . . .`
	`X        DB   123H`

Table A-4. (continued)

Message	Meaning
Double defined label	ASM-86. An identifier used as a label is used elsewhere in the program as a label or variable name. Example: `LAB3:  MOV   BX,5` `. . .` `LAB3:  CALL  MOVE`
Double defined symbol - treated as undefined	ASM-86. The identifier used as the name of an EQU directive is used as a name elsewhere in the program.
ERROR: BAD PARAMETER	PIP. An illegal parameter has been entered in a PIP command. Retype the entry correctly.
ERROR: CLOSE FILE - {filespec}	PIP. An output file cannot be closed. You should take appropriate action after checking to see if the correct disk is in the drive and that the disk is not write protected.
ERROR: DISK READ - {filespec}	PIP. The input disk file specified in a PIP command could not be read properly. This is usually the result of an unexpected end of file. Correct the problem in your file.
ERROR: DISK WRITE - {filespec}	PIP. A disk write operation could not be successfully performed during a PIP command, probably due to a full disk. You should either erase some unnecessary files or get another disk with more space and execute PIP again.

Table A-4. (continued)

Message	Meaning
ERROR: FILE NOT FOUND - {filespec}	PIP. An input file that you have specified does not exist.
ERROR: HEX RECORD CHECKSUM - {filespec}	PIP. A hex record checksum was encountered during the transfer of a hex file. The hex file with the checksum error should be corrected, probably by recreating the hex file.
Error in codemacro building	ASM-86. Either a codemacro contains invalid statements, or a codemacro directive was encountered outside a codemacro.
ERROR: INVALID DESTINATION	PIP. The destination specified in your PIP command is illegal. You have probably specified an input device as a destination.
ERROR: INVALID FORMAT	PIP. The format of your PIP command is illegal. See the description of the PIP command.
ERROR: INVALID HEX DIGIT - {filespec}	PIP. An invalid hex digit has been encountered while reading a hex file. The hex file with the invalid hex digit should be corrected, probably by recreating the hex file.
ERROR: INVALID SEPARATOR	PIP. You have placed an invalid character for a separator between two input filenames.

Table A-4. (continued)

Message	Meaning
ERROR: INVALID SOURCE	PIP. The source specified in your PIP command is illegal. You have probably specified an output device as a source.
ERROR: INVALID USER NUMBER	PIP. You have specified a User Number greater than 15. User Numbers are in the range 0 to 15.
ERROR: NO DIRECTORY SPACE - {filespec}	PIP. There is not enough directory space for the output file. You should either erase some unnecessary files or get another disk with more directory space and execute PIP again.
ERROR: QUIT NOT FOUND	PIP. The string argument to a Q parameter was not found in your input file.
ERROR: START NOT FOUND	PIP. The string argument to an S parameter could not be found in the source file.
ERROR: UNEXPECTED END OF HEX FILE - {filespec}	PIP. An end of file was encountered prior to a termination hex record. The hex file without a termination record should be corrected, probably by recreating the hex file.
ERROR: USER ABORTED	PIP. You aborted a PIP operation by pressing a key.

Table A-4. (continued)

Message	Meaning
ERROR: VERIFY - {filespec}	PIP. When copying with the V option, PIP found a difference when rereading the data just written and comparing it to the data in its memory buffer. Usually this indicates a failure of either the destination disk or drive.
File exists	You have asked Concurrent CP/M-86 to create a new file using a file specification that is already assigned to another file. Either delete the existing file or use another file specification.
File name syntax error	ASM-86. The filename in an INCLUDE directive is improperly formed. Example: `INCLUDE FILE.A86X`
File not found	Concurrent CP/M-86 could not find the specified file. Check that you have entered the correct drive specification or that you have the correct disk in the drive.
Garbage at end of line - ignored	ASM-86. Additional items were encountered on a line when ASM-86 was expecting an end of line. Examples: `NOLIST 4` `MOV AX,4 RET`

Table A-4. (continued)

Message	Meaning

Illegal expression element	ASM-86. An expression is improperly formed. Examples:
	`X DB 12X`
	`DW (4 * )`

Illegal first item	ASM-86. The first item on a source line is not a valid identifier, directive or mnemonic. Example: `1234H`
Illegal IF operand--IF ignored	ASM-86. Either the expression in an IF statement is not numeric, or it contains a forward reference.
Illegal pseudo instruction	ASM-86. Either a required identifier in front of a pseudo instruction is missing, or an identifier appears before a pseudo instruction that does not allow an identifier.

Illegal pseudo operand	ASM-86. The operand in a directive is invalid. Examples:
	`X EQU 0AGH`
	`TITLE UNQUOTED STRING`

Instruction not in code segment	ASM-86. An instruction appears in a segment other than a CSEG.

Table A-4. (continued)

Message	Meaning
Insufficient memory	DDT-86. There is not enough memory to load the file specified in an R or E command.
Invalid Assignment	STAT. An invalid device was specified in a STAT device assignment. Use the STAT val: display to list the valid assignments for each of the four logical STAT devices: CON:, RDR:, PUN: and LST:.
Label out of range	ASM-86. The label referred to in a call, jump or loop instruction is out of range. The label can be defined in a segment other than the segment containing the instruction. In the case of short instructions (JMPS, conditional jumps and loops), the label is more than 128 bytes from the location of the following instruction.
Memory request denied	DDT-86. A request for memory during an R command could not be fulfilled. Up to eight blocks of memory can be allocated at a given time.
Missing instruction	ASM-86. A prefix on a source line is not followed by an instruction. Example: `REPNZ`
Missing pseudo instruction	ASM-86. The first item on a source line is a valid identifier and the second item is not a valid directive that can be preceded by an identifier. Example: THIS IS A MISTAKE

Table A-4. (continued)

Message	Meaning

Missing segment information in operand	ASM-86. The operand in a CALLF or JMPF instruction (or an expression in a DD directive) does not contain segment information. The required segment information can be supplied by including a numeric field in the segment directive as shown:
	`        CSEG   1000H`
	`X:`
	`        . . .`
	`        JMPF   X`
	`        DD     X`

Missing type information in operand(s)	ASM-86. Neither instruction operand contains sufficient type information. Example: `MOV    [BX],10`
Nested IF illegal--IF ignored	ASM-86. The maximum nesting level for IF statements has been exceeded.
Nested INCLUDE not allowed	ASM-86. An INCLUDE directive was encountered within a file already being included.
No file	Concurrent CP/M-86 could not find the specified file, or no files exist. ASM-86. The indicated source or include file could not be found on the indicated drive. DDT-86. The file specified in an R or E command could not be found on the disk.

Table A-4. (continued)

Message	Meaning
No matching IF for ENDIF	ASM-86. An ENDIF statement was encountered without a matching IF statement.
No space	DDT-86. There is no space in the directory for the file being written by a W command.

Operand(s) mismatch instruction	ASM-86. Either an instruction has the wrong number of operands, or the types of the operands do not match. Examples:
	`MOV CX,1,2`
	`X DB 0`
	`MOV AX,X`

Parameter error	ASM-86. A parameter in the command tail of the ASM-86 command was specified incorrectly. Example: `ASM86 TEST $S;`
Symbol illegally forward referenced - neglected	ASM-86. The indicated symbol was illegally forward referenced in an ORG, RS, EQU or IF statement.
Symbol table overflow	ASM-86. There is not enough memory for the symbol table. Either reduce the length and/or number of symbols, or reassemble on a system with more memory available.

Table A-4. (continued)

Message	Meaning

Undefined element of expression	ASM-86. An identifier used as an operand is not defined or has been illegally forward referenced. Examples:
	`      JMP  X`
	`A     EQU  B`
	`B     EQU  5`
	`      MOV  AL,B`

Undefined instruction	ASM-86. The item following a label on a source line is not a valid instruction. Example: `DONE:  BAD   INSTR`
Use: [size] [ro] [rw] [sys] or [dir]	STAT. This message results from an invalid set file attributes command. These are the only options valid in a STAT filespec [option] command.
Use: STAT d:=RO	STAT. An invalid STAT drive command was given. The only valid drive assignment in STAT is STAT d:=RO.
Too Many Files	STAT. A STAT wildcard command matched more files in the directory than STAT can sort. STAT can sort a maximum of 512 files.
Verify error at s:o	DDT-86. The value placed in memory by a Fill, Set, Move, or Assemble command could not be read back correctly, indicating bad user memory or attempting to write to ROM or nonexistent memory at the indicated location.

End of Appendix A

Appendix B
User's Glossary

ambiguous filename: Filename that contains either of the Concurrent CP/M-86 wildcard characters, ? or *, in the primary filename or the filetype or both. When you replace characters in a filename with these wildcard characters, you create an ambiguous filename and can easily reference more than one Concurrent CP/M-86 file in a single command line.

applications program: Program that needs an operating system to provide an environment in which to execute. Typical applications programs are business accounting packages, word processing (editing) programs, mailing list programs, etc.

archive attribute: File attribute that indicates whether or not the file has been backed up. When you use PIP with the Archive option, it turns the archive attribute on. When a program changes a file, Concurrent CP/M-86 turns the archive attribute off, indicating that the file is new, and not backed up.

argument: Symbol, usually a letter, indicating a place into which you can substitute a number, letter, or name to give an appropriate meaning to the formula in question.

ASCII: The American Standard Code for Information Interchange is a standard code for representation of numbers, letters, and symbols. An ASCII text file is a file that can be intelligibly displayed on the video screen or printed on paper.

attribute: File characteristic that can be set to on or off.

background/foreground: A virtual console is switched-in to the foreground, appearing on your screen, or switched-out to the background, with output invisible.

back-up: Copy of a diskette or file made for safekeeping, or the creation of this diskette or file.

bit: A switch in memory that can be set to on (1) or off (0). Eight bits grouped together comprise a byte.

block: Area of memory or diskette reserved for a specific use.

boot, bootstrap: Process of loading an operating system into memory. Bootstrap procedures vary from system to system. The boot for an operating system must be customized for the memory size and hardware environment that the operating system will manage. Typically, the boot is loaded automatically and executed at power up or when the computer is reset. Sometimes called a cold start.

buffer: Area of memory or diskette that temporarily stores data during the transfer of information.

Buffered mode: A switched-out operating mode where program output does not appear on your screen but is toredin a buffer. Contrast with Physical mode and Dynamic mode.

built-in commands: Commands that permanently reside in memory. They respond quickly because they are not accessed from a diskette.

byte: Unit of memory or diskette storage containing eight bits.

command: Elements of a Concurrent CP/M-86 command line. In general, a Concurrent CP/M-86 command has three parts: the command keyword, the command tail, and a carriage return.

command file: Series of coded machine executable instructions stored on diskette as a program file, invoked in Concurrent CP/M-86 by typing the command keyword next to the system prompt on the console. The Concurrent CP/M-86 command files generally have a filetype of CMD. Files are either command files or data files. Same as a command program.

command keyword: Name that identifies a Concurrent CP/M-86 command, usually the primary filename of a file of type CMD, or the name of a queue associated with a Resident System Process. The command keyword precedes the command tail and the carriage return in the command line.

command syntax: Statement that defines the correct way to enter a command. The correct structure generally includes the command keyword, the command tail, and a carriage return. A syntax line usually contains symbols that you should replace with actual values when you enter the command.

command tail: Part of a command that follows the command keyword in the command line. The command tail can include a drive specification, a filename and/or filetype, a password, and options or parameters. Some commands do not require a command tail.

concatenate: Term that describes one of PIP's operations that copies two or more separate files into one new file in the specified sequence.

console: Primary input/output device. The console consists of a listing device such as a screen and a keyboard through which the user communicates with the operating system or applications program. Under Concurrent CP/M-86, a virtual console is a channel that is capable of initiating programs.

control character: Nonprinting character combination that sends a simple command to the currently executing process. To enter a control character, hold down the control key on your terminal and strike the character key specified. In this document, the control key is represented by CTRL. A CTRL-X, for example, erases the command line. See Appendix D.

cursor: One-character symbol that can appear anywhere on the console screen. The cursor indicates the position where the next keystroke at the console will have an effect.

data file: Nonexecutable collection of similar information that generally requires a command file to manipulate it.

default: Currently selected diskette drive, user number, password, console, or list device. Any command that does not specify a diskette drive or a user number references the default diskette drive and user number. When Concurrent CP/M-86 is first invoked, the default drive is the system drive. The default user number varies from console to console, until changed with the USER command. A default display is a display generated by a command keyword without any options.

delimiter: Special characters that separate different items in a command line. For example, in Concurrent CP/M-86, a colon separates the drive specification from the filename. A period separates the filename from the filetype. A semicolon separates the filename and type from the password, and square brackets separate any options from their command or file specification. Commas separate one item in an option list from another. All of the above special characters are delimiters.

directory: Portion of a diskette that contains entries for each file on the disk. In response to the DIR command, Concurrent CP/M-86 displays the filenames stored in the directory.

DIR attribute: File attribute that causes a file to be accessible from the default user number and drive only.

directory label: Same as label. See **label.**

disk, diskette: Magnetic media used to store information. Programs and data are recorded on the diskette in the same way that music is recorded on a cassette tape. The term diskette refers to smaller capacity removable floppy diskettes. Disk can refer to a diskette, a removable cartridge, or a fixed hard disk.

disk drive, diskette drive: Peripheral device that reads and writes on hard or floppy disks. Concurrent CP/M-86 assigns a letter to each drive under its control. For example, Concurrent CP/M-86 can refer to the drives in a four-drive system as A, B, C, and D.

drive label: Same as label. See **label.**

Dynamic mode: A switched-out operating mode where program output is not saved on a diskette file. All but the last screenful of output is lost. Contrast with Buffered mode and Physical mode.

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 Concurrent CP/M-86 editor is invoked by typing the command ED next to the system prompt on the console. (See ED in Section 5 of this manual).

executable: Ready to be run by the computer. Executable code is a series of instructions that can be carried out by the computer. For example, the computer cannot execute names and addresses, but it can execute a program that prints all those names and addresses on mailing labels.

FCB: File Control Block.

field: Portion of a record containing one data item such as a person's name, an address, or a phone number.

file: Collection of characters, instructions, or data stored on a disk. The user can create files on a diskette.

File Control Block: Structure used for accessing files on diskette. Contains the drive, filename, and filetype of a file to be accessed or created on the diskette.

filename: Name assigned to a file. A filename can include a primary filename of 1-8 characters and a filetype of 0-3 characters. A period separates the primary filename from the filetype.

file specification: Unique file identifier. A complete Concurrent CP/M-86 file specification includes a diskette drive specification followed by a colon (d:), a primary filename of 1 to 8 characters, a period and a filetype of 0 to 3 characters, a semicolon and a password. For example, b:example.tex;password is a complete Concurrent CP/M-86 file specification.

filetype: Extension to a filename. A filetype can be from 0 to 3 characters and must be separated from the primary filename by a period. A filetype can tell something about the file. Certain programs require that files to be processed have certain filetypes (see Appendix E).

floppy diskette: Flexible magnetic diskette used to store information. Floppy diskettes come in 5 1/4 and 8 inch diameters.

foreground/background: A virtual console is switched-in to the foreground, appearing on your screen, or switched-out to the background, with output invisible.

global option: A use of parameters, delimited by square brackets, to effect more than one file in a directory or on a drive. Contrast with **local option.**

hard disk: Rigid, platter-like, magnetic disk sealed in a container. A hard disk stores more information than a floppy diskette.

hardware: Physical components of a computer.

hex file: ASCII-printable representation of a command (machine language) file.

hexadecimal notation: Notation for the base 16 number system using the symbols 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F to represent the sixteen digits. Machine code is often converted to hexadecimal notation because it can be easily represented by ASCII characters and therefore printed on the console screen or on paper. In this manual we follow the industry standard of prefacing hexadecimal numbers with 0 and ending them with an "H": 0FFH is hexadecimal FF.

input: Data going into the computer, usually from an operator typing at the terminal or by a program reading from the diskette.

interface: Object that allows two independent systems to communicate with each other, as an interface between hardware and software in a microcomputer.

I/O: Abbreviation for input/output.

keyword: See **command keyword.**

kilobyte: 1024 bytes denoted as 1K. 32 kilobytes equal 32K. 1024 kilobytes equal one megabyte, or over one million bytes.

label: Entry in the directory. The optional label contains information that describes special attributes of the diskette to the operating system. For example, the label tells Concurrent CP/M-86 whether or not time stamping and password protection are turned on for that diskette. You can give a label a name to help identify the data that is sorted on a given diskette.

list device: Device, such as a printer, onto which data can be listed or printed.

local option: A use of parameters, delimited by square brackets, to affect a single file in a directory or on a drive. Contrast with **global option.**

logged in: Made known to the operating system, in reference to drives. A drive is logged in when it is selected by the user or an executing process and remains selected or logged in until a DSKRESET is performed or the whole operating system is reset.

logical: Representation of something that might or might not be the same in its actual physical form. For example, a hard disk can occupy one physical drive, and yet you can divide the available storage on it to appear to the user as if it were in several different drives. These apparent drives are the logical drives.

megabyte: Over one million bytes; 1024 kilobytes (see byte).

microprocessor: Silicon chip that is the Central Processing Unit (CPU) of the microcomputer.

multiprogramming: Capability of the operating system to coordinate the execution of more than one program at a time.

multiuser: Ability of an operating system to support more than one independent user initiating different programs at the same time.

operating system: Collection of programs that supervises the running of other programs and the management of computer resources. An operating system provides an orderly input/output environment between the computer and its peripheral devices. It enables user-written programs to execute safely.

option: One of many variables that can be appended to a command.

output: Data that the processor sends to the console, diskette or printer.

parameter: Value in the command tail that provides additional information for the command. Technically, a parameter is a required element of a command.

password: User-created extension to a filename that enables the user to add extra protection to his files. The password may then be required to access that file. A password can be up to eight characters long and include any numeric or upper- or lower-case characters and some special characters.

peripheral devices: Devices external to the CPU. For example, terminals, printers, and diskette drives are common peripheral devices that are not part of the processor, but are used in conjunction with it.

physical: Actual hardware of a computer. The physical environment varies from computer to computer.

Physical mode: A switched-in operating mode in which program output appears on your screen as the program generates it. Contrast with Dynamic and Buffered modes.

primary filename: First 8 characters of a filename. The primary filename is a unique name that helps the user identify the file contents. A primary filename contains 1 to 8 characters and can include any letter or number and some special characters. The primary filename follows the optional drive specification and precedes the optional filetype.

process: When a program is actually executing, as opposed to being in a static state of storage on diskette, it is called a process.

program: A series of specially coded instructions that performs specific tasks when executed by a computer.

prompt: Any characters displayed on the video screen to help the user decide what the next appropriate action is. A system prompt is a special prompt displayed by the operating system. The system prompt indicates to the user that the operating system is ready to accept input. The Concurrent CP/M-86 system prompt is two characters followed by an angle bracket. The first character is numeric and indicates the default user number. The second character is alpha and indicates the default drive. Some applications programs have their own special system prompts.

queue: First in, first out list. Under Concurrent CP/M-86, a queue is treated as a file in memory. For example, a queue has a Queue Control Block instead of a File Control Block. In general, processes use queues to pass information to other processes. One of the functions of a queue is to wake up a Resident System Process and pass your command to it. Refer to the Concurrent CP/M-86 Operating System Programmer's Guide for further discussion.

Read-Only: Attribute that can be assigned to a diskette file or a diskette drive. When assigned to a file, the Read-Only attribute allows you to read from that file but not write any changes to it. When assigned to a drive, the Read-Only attribute allows you to read any file on the diskette, but prevents you from adding a new file, erasing or changing a file, renaming a file, or writing on the disk. The SET and STAT commands can set a file or a drive to Read-Only. Every file and drive is either Read-Only or Read-Write. The default setting for drives and files is Read-Write, but an error in setting diskette density or resetting the diskette or other error found by Concurrent CP/M-86 automatically sets the drive to Read-Only until the error is corrected. Files and diskette drives may be set to either Read-Only or Read-Write.

Read-Write: Attribute that can be assigned to a diskette file or a diskette drive. The Read-Write attribute allows you to read from and write to a specific Read-Write file or to any file on a diskette that is in a drive set to Read-Write. A file or drive can be set to either Read-Only or Read-Write.

record: Collection of data. A file consists of one or more records stored on diskette. A Concurrent CP/M-86 record is 128 bytes long.

Resident System Process: A process that is made part of Concurrent CP/M-86 during system generation. An RSP can be associated with a queue that can be accessed by a command keyword. The file containing the code may be stored on diskette as a file of type RSP.

R/O: Abbreviation for Read-Only.

RSP: Resident System Process.

run a program: Start a program executing. When a program is running, the computer is executing a sequence of instructions.

R/W: Abbreviation for Read-Write.

sector: Portion of a diskette track. There are a specific number of sectors on each track.

software: Specially coded programs transmit machine readable instructions to the computer, as opposed to hardware, which is the actual physical components of a computer.

source file: ASCII text file that is an input file for a processing program, such as an editor, text formatter, assembler or compiler.

syntax: Format for entering a given command.

system attribute: Attribute assigned to a file enabling that file to be accessed from other than the default drive and user number. System files are generally placed on the system drive in user 0 because they are most easily and efficiently accessed from that location. A file can have either the SYS attribute or the DIR attribute.

system console: Same as console. A console that displays a Concurrent CP/M-86 system prompt.

system drive: Drive on which Concurrent CP/M-86 looks for files with the SYS attribute after checking the default or specified drive.

systems program: Program that contributes to the operating system package.

system prompt: Symbol displayed by the operating system indicating that the system is ready to receive input. See prompt.

terminal: Generally, the same as a console.

time stamp: Record of when a file was created, accessed, or updated. In Concurrent CP/M-86, the SET command turns the time stamp on. The time and date information is appended to a file in the XFCB. The time stamps are displayed by the SDIR command.

TMP: Terminal Message Processor; a part of Concurrent CP/M-86 that determines the user interface, returns certain error messages, etc. See the Concurrent CP/M-86 Operating System Programmer's Guide for further details.

track: Concentric rings dividing a diskette. There are 77 tracks on a typical single-density eight-inch floppy diskette.

turn-key application: Application designed for the noncomputer-oriented user. For example, a typical turn-key application is designed so that the operator needs only to turn on the computer, insert the proper program diskette and select the desired procedure from a selection of functions (menu) displayed on the screen.

upward-compatible: Term meaning that a program created for the previously released operating system (or compiler, etc.) runs under the newly released version of the same operating system.

user number: Number assigned to a region of the diskette directory so that different processes need only deal with their own files and have their own directories even though they are all working from the same diskette. In Concurrent CP/M-86, there can be up to sixteen users on a single diskette.

utility: Tool program that enables the user to perform certain operations, such as copying files, erasing files, and editing files. Utilities are created for the convenience of programmers and users. Concurrent CP/M-86 is distributed with over 25 utilities.

virtual console: The computer console that is switched-in, appearing on your screen, or switched-out and invisible. Programs run on individual virtual consoles regardless of whether they appear on the Personal Computer screen or not. See Section 1.1.1 for a description of how virtual consoles work.

wildcard characters: Special characters that match certain specified items. In Concurrent CP/M-86 there are two wildcard characters, ? and *. The ? can be substituted for any single character in a filename, and the * can be substituted for the primary filename or the filetype or both. By placing wildcard characters in filenames, the user creates an ambiguous filename and can quickly reference one or more files.

XFCB: Extended File Control Block. XFCBs store passwords and time and date stamping of files. See FCB and Block in the Concurrent CP/M-86 Operating System Programmer's Guide.

End of Appendix B

Appendix C
Concurrent CP/M-86 Command Summary

Concurrent CP/M-86 is distributed with over 25 utilities. Table C-1 lists the utilities described in this manual alphabetically. The required parts of the command line are printed in boldface.

Table C-1. Command Summary

Syntax	Definition and Examples

ABORT programname n	ABORT stops execution of program programname initiated from console n.
	
	`0A>ABORT PIP`
	`2B>ABORT SDIR 2`

ASM86 filename.A86	ASM86 invokes ASM-86 Assembler and processes the specified file. The file must be an assembly language file of .A86 filetype. (See the Concurrent CP/M-86 Operating System Programmer's Guide for the details of assembly language programming and running ASM-86.) `0A>ASM86 filemgr.A86`
CONFIG printer# baudrate wordlength parity stopbits	CONFIG sets operating parameters for serial ports. `0A>CONFIG P3 9600 7 NONE 1`
DDT86 filespec	DDT86 invokes the DDT-86 Debugging Tool. (See the Concurrent CP/M-86 Operating System Programmer's Guide for further details of assembly language program troubleshooting with DDT-86.) `0A>DDT86 filemgr.cmd`

Table C-1. (continued)

Syntax	Definition and Examples

DIR filespec [SYS,Gn], filespec	DIR displays a directory of files on a diskette. Accepts ambiguous filenames to display a group of similarly named files. The [SYS] option displays system files also. The [Gn] option displays the directory of user number n.
	
	`0A>DIR`
	`0A>DIR B:`
	`0A>DIR B:DRAFT.TXT`
	`0A>DIR B:*.TXT [SYS]`
	`0A>DIR B:DRAFT.*`

DSKMAINT	DSKMAINT performs diskette verification, formatting, and copying of entire diskettes. Use PIP to copy single files. `0A>DSKMAINT`

DSKRESET d:,d:,d:	DSKRESET logs out all or specified drives except for the default drive. This will reinitialize the drives the next time they are accessed. Done before diskette changes.
	
	`0A>DSKRESET`
	`0A>DSKRESET A:,B:,C:`

ED filespec	ED creates or edits programs and data files. Does not accept ambiguous filenames.
	
	`0A>ED DRAFT.TXT`
	`0A>ED B:DRAFT.TXT`
	`0A>ED F:DOCUMENT.LAW;SECRET`

Table C-1. (continued)

Syntax	Definition and Examples

ERA filespec [XFCB]	ERA erases a file or a group of files. Accepts ambiguous filenames. With [XFCB] it erases only XFCBs of specified files.
	
	`0A>ERA DRAFT.BAK`
	`0A>ERA B:*.BAK`
	`0A>ERA B:DRAFT.*;SECRET`
	`0A>ERA B:*.*`
	`0A>ERA B:DRAFT.* [XFCB]`

ERAQ filespec [XFCB]	ERAQ is the same as ERA except it queries for each specified file before erasing.
	
	`0A>ERAQ F:*.*[XFCB]`
	`3F>ERAQ *.CMD;PASSWORD`
	`2C>ERAQ D:DRAFT.*;SECRET`

FUNCTION filespec	FUNCTION lets you assign functions to function and numeric keys. `0A>FUNCTION data.pfk`
GENCMD filespec (8080 CODE[An,Bn,Mn,Xn] DATA [An,Bn,Mn,Xn] STACK[An,Bn,Mn,Xn] EXTRA [An,Bn,Mn,Xn])	GENCMD produces an executable .CMD file usable as a transient utility command, such as TOD.CMD, PIP.CMD, or STAT.CMD. `0A>GENCMD MYFILE`

Table C-1. (continued)

Syntax	Definition and Examples
HELP	HELP provides an on-line summary of Concurrent CP/M-86 commands and their syntax. `0A>HELP topic subtopics`

PIP destination filespec [Gn]=source filespec [options]	PIP transfers information between peripheral devices and concatenates files.
	`0A>PIP`
	`0A>PIP B:DRAFT.TXT=A:`
	`0A>PIP LST:=B:DRAFT.TXT;PASSWORD`
	`2B>PIP PRN:=A:DRAFT1.TXT[T8]`
	`3C>PIP B:ABC.TXT;SECRET[G0]=DEF.TXT;PASS`
	`4F>PIP B:ABC.TXT=DRAFT.TXT;PASSWORD[G0]`
	`0F>PIP B:=*.*[AV]`

PRINTER n	PRINTER displays the printer number for your console. With a number n, it sets the printer number to n for your console.
	`0A>PRINTER`
	`3F>PRINTER 2`

REN destination filespec = source filespec	REN renames a file without changing its contents. Accepts ambiguous filenames or filetypes if present in both source and destination.
	`0A>REN DRAFT.TXT = FIRST.TXT`
	`0A>REN B:DRAFT.TXT = B:FIRST.TXT`
	`6D>REN DRAFT5.TXT = E:DRAFT.TXT;PASSWORD`
	`0B>REN *.TEX = *.WRK`

Table C-1. (continued)

Syntax	Definition and Examples
SDIR [options] filespec, filespec	SDIR displays the diskette directory with options. 1A>SDIR 2A>SDIR B: 3B>SDIR [DRIVE=ALL, USER=ALL, SIZE] 4C>SDIR [XFCB, DRIVE=(a,b,c)] 5E>SDIR [USER=2, RW] *.TEX, *.WRK

SET [HELP]
SET d:[NAME=diskname]
SET d:[PASS=password, PROTECT=ON, DEFAULT=password]
SET d:[RO,RW]
SET filespec [PASS=password,TIME=ON]
SET filespec [PROTECT=READ,PROTECT=WRITE, PROTECT=DELETE,PROTECT=NONE]
SET filespec [RW, RO, DIR, SYS]
	SET controls password protection, date/time stamps, and file or drive attributes. SET commands effect either an entire drive, a group of files, or a single file. (See Section 4.20).
	
	0A>SET D:[RO]
	0C>SET *.CMD [RO,SYS], *.TXT [RO,SYS]
	1B>SET *.CMD [SYS,RO,PASS=SECRET, PROT=READ]
	2C>SET *.CMD [RW,PROTECT=NONE,DIR]
	0A>SET B:[PASSWORD=SECRET]
	0A>SET [NAME=SYSTMDSK]
	0A>SET [PASS=<cr>
	0A>SET *.CMD [PASSWORD=SECRET]

SHOW option, option, option... options = SPACE, USERS, DRIVES, LABEL, HELP	SHOW displays amount of free diskette space, the drive label status, the active user numbers on a drive, and the drive characteristics. 0A>SHOW 0B>SHOW C: 1C>SHOW DRIVES 2D>SHOW USERS 3E>SHOW LABEL 3E>SHOW E:,F:,E:USERS,F:USERS

Table C-1. (continued)

Syntax	Definition and Examples

STAT d:=RO
STAT d:DSK:
STAT d:USR:
STAT VAL:
STAT filespec [attribute]
attributes = RO, RW, SYS, DIR or SIZE
	STAT provides information about a file or a group of files on a diskette. Also assigns attributes to a file, diskette, or drive. The SIZE option displays the last record number, the number of records, bytes, and FCBs, and all the attributes of a file.
	
	`0A>STAT`
	`0A>STAT B:`
	`1F>STAT D:*.CMD`

SUBMIT filespec [actual parameters]	SUBMIT submits a batch process consisting of a file of Concurrent CP/M-86 commands (one command per line in file). The filename must be a file of type SUB. Parameters following the filename are substituted for their corresponding parameters in the file. (See Section 4.23.)
	
	`0A>SUBMIT START`
	`0A>SUBMIT B:START`
	`0A>SUBMIT START B LETTER`
	`1F>SUBMIT B:START`

SYSDISK
SYSDISK d:
	SYSDISK without a drive number returns the current system diskette. SYSDISK with a drive number d: sets that diskette to be the system diskette.
	
	`0A>SYSDISK M:`
	
	`System disk is M:`

Table C-1. (continued)

Syntax	Definition and Examples
TOD P **TOD mm/dd/yy hh:mm:ss**	TOD displays the system date and time. With a date and time, TOD sets the system time to the date and time specified. The P option displays date and time continuously. 0A>**TOD** 1B>**TOD P** 1C>**TOD 02/14/82 12:01:00**
TYPE filespec [PAGE]	TYPE displays contents of a file containing ASCII-coded information. Does not accept ambiguous filenames. The [PAGE] option pauses after each screen (24 lines) is displayed until you strike a key. [Pn] specifies the number of lines per screen. 0A>**TYPE DRAFT.TXT** 0A>**TYPE B:DRAFT.TXT [PAGE]** 1F>**TYPE E:DRAFT.TXT;PASSWORD[P15]**
USER n	USER changes the current user number. Sets the user number to n, where n is an integer from 0-15. 0A>**USER 8**
VCMODE **VCMODE** help **VCMODE** dynamic **VCMODE** buffered **VCMODE** size=n	VCMODE sets virtual consoles to Buffered or Dynamic mode, and specifies maximum buffer file size where n=1 - 8191. 0A>**VCMODE** 0A>**VCMODE HELP** 0A>**VCMODE DYNAMIC** 0A>**VCMODE BUFFERED** 0A>**VCMODE SIZE=16**

Appendix D
Concurrent CP/M-86 Control Character Summary

Table D-1. Concurrent CP/M-86 Control Characters

Keystroke	Action
←	moves cursor back one space, erases previous character.
CTRL-C	prompts to abort a process currently running at a given console.
CTRL-E	forces a physical carriage return, but does not send command to Concurrent CP/M-86.
CTRL-F	flush Virtual Console disk buffer
CTRL-H	same as BACKSPACE.
CTRL-J	line feed, terminates input at the console.
↵	carriage return.
CTRL-M	same as carriage return.
CTRL-P	echoes all console activity at the printer; a second CTRL-P ends printer echo. This only works if your system is connected to a printer.
CTRL-Q	resumes console display after CTRL-S.
CTRL-R	retypes current command line; useful after using RUB or DEL key.
CTRL-S	stops console display temporarily; CTRL-Q resumes the listing.
CTRL-U	cancels line, displays #, cursor moves down one line and awaits a new command.
CTRL-X	deletes all characters in command line.
CTRL-Z	string separator for PIP and ED; terminates console input when console is used as a source device with PIP.

If you have used regular CP/M-86 on your Personal Computer, you will remember that pressing ALT then any character code number on the decimal keypad generates that character on your screen. Concurrent CP/M-86 works differently. Pressing the keys generates the first 128 characters in the character set; pressing ALT-<key> generates the next 128 characters.

There are other differences between how keys function in CP/M-86 and Concurrent CP/M-86, shown in the following table.

Table D-2. Concurrent CP/M-86 Control Keys

Keystroke	Action
PrtSc	In Concurrent CP/M-86, pressing PrtSc generates a CTRL-P.
Scroll Lock	In Concurrent CP/M-86, pressing Scroll Lock generates a CTRL-S.
<Shift>Scroll Lock	In Concurrent CP/M-86, pressing the Shift key, ⇧ , while pressing Scroll Lock generates a CTRL-Q.
<CTRL>Scroll Lock <CTRL>Break	In Concurrent CP/M-86, pressing the CTRL key while pressing the Scroll Lock key generates a CTRL-C.
<CTRL> ESC	In Concurrent CP/M-86, pressing the CTRL key while pressing the ESC key clears the beep buffer.
←	In Concurrent CP/M-86, pressing the ← key generates a CTRL-H (backspace).
<Shift> ←	In Concurrent CP/M-86, pressing the Shift ← also generates a CTRL-H (backspace).
<CTRL> ←	In Concurrent CP/M-86, pressing CTRL-← generates an ASCII delete.

End of Appendix D

Appendix E
Filetypes

Concurrent CP/M-86 identifies every file by a unique file specification, which consists of a drive specification, a filename, a filetype, and an optional password. The filetype is an optional three character ending separated from the filename by a period. The filetype generally indicates a special kind of file. The following table lists common filetypes and their meanings.

Table E-1. Filetypes

Filetype	Indication
A86	Assembly language source file; the Concurrent CP/M-86 assembler, ASM-86, assembles or translates a file of type .A86 into machine language.
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 diskette as the back-up file, so you can refer to it.
CMD	Command file that contains instructions in machine executable code.
H86	Program file in hexadecimal format.
LST	Printable file that can be displayed on a console or printer.
PRN	Printable file that can be displayed on a console or printer.
SUB	Filetype required for SUBMIT program containing one or more Concurrent CP/M-86 commands. The SUBMIT program executes the commands in the submit file providing a batch mode for Concurrent CP/M-86.
TXT	Text file.
$$$	Temporary file.

End of Appendix E

Appendix F
Checklist For Using Files

- If the file is set to Read-Only, you can read the file but you cannot write to the file.
- If the drive is set to Read-Only, you cannot write to files on that drive. This occurs if you forget to use DSKRESET after changing a diskette.
- If you have typed a CTRL-S or pressed the SCROLL LOCK key, your keyboard is locked until you type a CTRL-Q to unlock it.
- Files with the DIR attribute can be accessed only if they are in the default user area on the default or specified drive except when using DIR or PIP with the G option.
- Files with the SYS attribute can be accessed if they are in the default user area or user 0 of the default or specified drive, or in the default or user 0 area of the system drive.
- If a drive is specified, Concurrent CP/M-86 looks for a file only in the default and zero user areas of the specified drive.
- If the command line specifies a drive or a password, Concurrent CP/M-86 looks for a CMD file on diskette.
- If the file is password protected, you might get a password error message.
- Is the password protection mode set to READ, WRITE, DELETE, or NONE? (See the SET command in Section 4.20).
    - If the password protection mode is set to READ, then you need a password to read the file.
    - If the password protection mode is set to WRITE, you can read the file without supplying the password, but you need the password to write to the file.
    - If the password protection mode is set to DELETE, you can read or write to the file, but you need the password to erase it.
    - If the mode is set to NONE, the password is erased: you no longer need it at all.
- Does the drive have a label? (See the SET command in Section 4.20).
    - If the drive has a label and password protection is turned on for the drive, then you need a password to access any password protected files on that drive.

Appendix G
Drive and File Status Summary

Table G-1. Display File Size

Command	# of Bytes in File	# of Recs in File	Last Rec#	Free diskette Space	Totals
SDIR	X	X		X	X
SDIR [SIZE]	X				
STAT filespec	X	X		X	X
STAT filespec [SIZE]	X	X	X	X	X

Table G-2. Display File Attributes

Command	SYS or DIR	RW or RO	SYS Files displayed?	User #	Archive/ User Def. Attribute
DIR [SYS]			X		
DIR filespec [SYS]			X		
SDIR	X	X	X	X	X
STAT filespec		X	X		X
STAT filespec[SIZE]	X		X	X	X

Table G-3. Display Time Stamping and Protection Modes

Command	Creation or Last Access	Last Update	Password Mode	XFCB
SDIR	X	X	X	X

Table G-4. Ways to Display Diskette, Label, and System Status

Command	RO or RW	Free Diskette Space	Drive Display	Active Users	Active Files	Label Display
SHOW	X	X				
SHOW d:	X	X				
SHOW SPACE	X	X				
SHOW DRIVE			X			
SHOW USERS				X	X	
SHOW LABEL						X
STAT	X	X				
STAT d:	X	X				
STAT USER				X	X	
STAT DSK:			X			

Sample SHOW DRIVE Display:

```
0A>SHOW DRIVE

       A: Drive Characteristics
   1,248: 128 Byte Record Capacity
     156: Kilobyte Drive  Capacity
      64: 32 Byte  Directory Entries
      64: Checked  Directory Entries
     128: Records / Directory Entry
       8: Records / Block
      32: Sectors / Track
       1: Reserved  Tracks

       B: Drive Characteristics
   1,248: 128 Byte Record Capacity
     156: Kilobyte Drive  Capacity
      64: 32 Byte  Directory Entries
      64: Checked  Directory Entries
     128: Records / Directory Entry
       8: Records / Block
      32: Sectors / Track
       1: Reserved  Tracks
```

Sample SHOW LABEL Display:

```
Directory     Passwds Make  Stamp  Stamp  Label   Created Label   Updated
Label         Req'd   XFCBs Create Update
------------- ------- ----- ------ ------ -------------- --------------
TOMSDISK.DAT    on      on    on     on   07/04/81 10:30 07/08/81 09:30
```

End of Appendix G

Appendix H
Console Escape Sequences

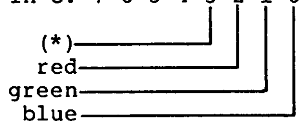
You can control the console's video attributes, such as cursor control, video blink, video intensity and so on, by sending character sequences to the console. The first character of each sequence is the ASCII Escape character, 01BH, so these sequences are called escape sequences. With minor exceptions, the escape sequences are a superset of those required by a DEC model VT-52 CRT terminal.

To set the cursor row and column position, you must send four characters to the console. The third and fourth characters represent the desired row and column positions respectively. On the display, rows are numbered 00H through 018H, and columns are numbered 00H through 04FH. To compute the proper ASCII characters to send to the console, add 020H to the desired row and column number. For example, to position the cursor at row 1 column 6, send the four ASCII characters ESC, Y, !, and &. In this example, the code for the ASCII character !, 021H, represents row 1, and the code for the ASCII character &, 026H, represents column 6.

To set the foreground or background color, send ASCII Escape, ASCII b for foreground, or c for background, and a binary number. The least significant 3 bits of the binary number specify the colors as follows: the least significant bit specifies blue, the next most significant bit specifies green, and the next most significant bit specifies red. Therefore, to set the foreground color to green, you must output ASCII Escape, ASCII b, and binary 0000 0010 (02H).

The escape sequences required to control the video display attributes are listed below.

Function	**Escape Sequence**
Cursor Home	ESC H
Cursor Forward	ESC C
Cursor Backward	ESC D
Cursor Down	ESC B
Cursor Up	ESC A
Reverse Index	ESC I
Save Cursor Position	ESC j
Restore Cursor Position	ESC k
Set Cursor Position	ESC Y (r)(c),r=row (0 through 018H)+020H c=column (0 through 4FH)+020H
Clear Display	ESC E
Erase Beginning of Display	ESC d
Erase to End of Page	ESC J
Erase Entire Line	ESC L
Erase Beginning of Line	ESC o
Erase to End of Line	ESC K
Insert Blank Line	ESC L
Delete Line	ESC M
Delete Character	ESC N
Set Foreground Color	ESC b (c) color and intensity defined
Set Background Color	ESC c (c) by bits in c: 7 6 5 4 3 2 1 0
Enable Cursor	ESC e
Disable Cursor	ESC f
Enter Reverse Video Mode	ESC p
Exit Reverse Video Mode	ESC q
Enter Blink Video Mode	ESC s
Exit Blink Video Mode	ESC t
Enter Intensify Mode	ESC r
Exit Intensify Mode	ESC u
Wrap at End of Line	ESC v
Discard at End of Line	ESC w
Set Mode COLOR	ESC x
Set Mode B&W	ESC y
Program Function Keys	ESC :



* Note: when you set foreground color, Bit 3 controls high-intensity. When you set background color, Bit 3 controls blinking.

End of Appendix H

Index