GETTING STARTED WITH

ROB PATTEN & PAUL CALANDRINO

COMPLETE INTRODUCTION TO CP/M-80 VERSION 2.2 WITH
STEP-BY-STEP EXAMPLES PLUS A
DETACHABLE COMMAND REFERENCE CARD

HAYDEN

GETTING STARTED WITH

CP/M®

GETTING STARTED WITH

ROB PATTEN & PAUL CALANDRINO

HAYDEN BOOK COMPANY, INC.
Rochelle Park, New Jersey

Production Editor: MAUREEN CONNELLY
Art Director: JIM BERNARD
Text Design: GANIS AND HARRIS INC., NEW YORK
Cover Design: JEANNETTE JACOBS
Compositor: PUBLISHER'S PHOTOTYPE, INC.
Printed and Bound by: BANTA COMPANY

Printed in the United States of America

1	2	3	4	5	6	7	8	9	PRINTING
83	84	85	86	87	88	89	90	91	YEAR

Previously published as *An Introduction to CP/M for the Beginning User*

CONTENTS

PREFACE

CP/M stands for Control *P*rogram for *M*icrocomputers, although sometimes it is called Control *P*rogram/*M*onitor, or Control *P*rogram for *M*icroprocessors. However translated, it is the same product. CP/M-80 is the most widely used operating system for 8-bit microcomputers. Most leading manufacturers of computer systems for business and home use are offering CP/M as a standard software component. CP/M's popularity as an operating system has prompted the development of an almost limitless number of CP/M-compatible application programs, and the variety of these programs is constantly increasing.

This book is designed to introduce the first-time user to CP/M-80 2.2. It offers an overview of how CP/M operates in the microcomputer environment and provides detailed descriptions of the structure and function of the CP/M commands most useful to a beginning user: those most often used for storing and retrieving information, inquiring into the status of a device or diskette, and running application programs for business or home use. You'll learn how to use these commands to your best advantage.

The beginning user should be aware that this guide does not make any attempt to teach you how to program. The more advanced user should also realize that the tutorial nature of this guide imposes limitations on our descriptions of those relatively complex features of CP/M, such as the ASM program. Where we have limited our discussion of a subject, references are made to other documents.

The presentation of information in this introductory guide assumes the reader has no previous experience with computers and no prior knowledge of programming. It has been written to provide the beginning user with easy-to-understand introductions to such subjects as:

- Handling and taking care of diskettes.
- Storing and transferring information from one location to another.
- Files: what they are, and how to create and name them.
- What to do when error messages appear on the screen.
- Specific command structures and options.

A GUIDE TO THE CONTENTS OF THIS BOOK

Chapter One: Provides a general overview of the way CP/M works in the microcomputer environment and describes the procedures and conventions for issuing CP/M commands. This chapter also contains information on diskettes: how data is stored on them, and how to handle and maintain them.

Chapter Two: Contains detailed descriptions of CP/M's line editing and built-in and transient commands, including basic structure, options or parameters associated with each command, and one or more examples and error messages for particular commands. The first section of this chapter presents a simplified view of CP/M's internal structure.

Chapter Three: Shows you how to prepare *work copies* of CP/M and other software, so that you are assured of always having an accurate copy of the original software.

Chapter Four: Discusses the procedures for running application programs under the control of CP/M.

Chapter Five: Contains directions and examples to lead the user through a sample session that employs some of the more commonly used CP/M commands.

Appendix A: The Command/Function Reference describes the basic functions of each command in a concise manner.

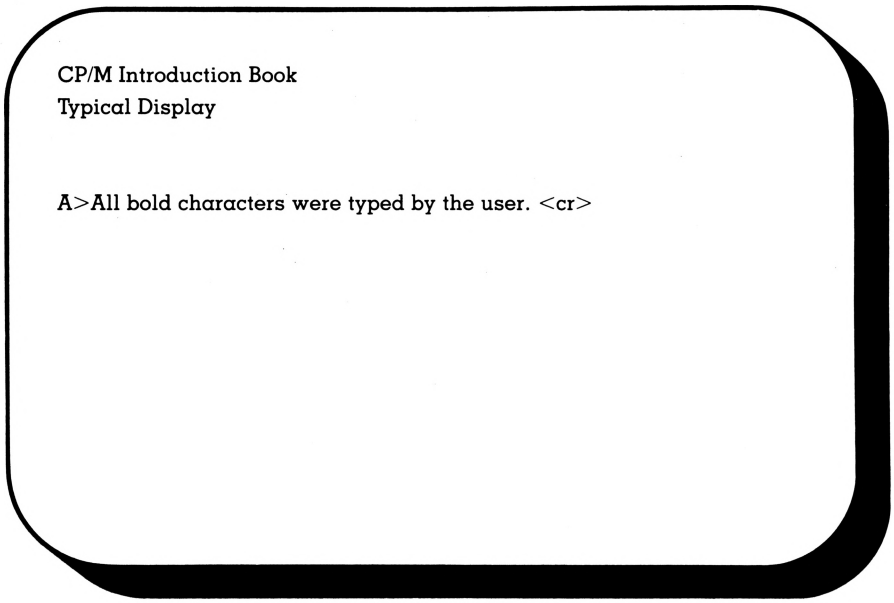
Appendix B: The Glossary defines terms which pertain to CP/M or to microprocessors in general. This glossary will be especially helpful to the first-time user, who should become familiar with these terms before starting with Chapter One.

Appendix C: We have included a list of references to other documents that will be useful to more advanced CP/M users and to beginning users who want to supplement their knowledge of CP/M.

Index listings are organized according to subject and function. For example, if you want to find a certain file on a diskette but don't know or remember which command to use, you can look in the index under "Searching for files" or "Files, searching for."

The Command Reference Card, which is easily detachable from this book, contains a complete list of CP/M command structures, specifying the options or parameters of each command.

It will be helpful for you to know about some of the conventions we use in this guide for the presentation of information. In many of our descriptions, we give you examples of what will appear on the screen of your computer. These examples are called *displays*. If user input (something the user types on the keyboard) is shown on the display, that input will appear in boldface, as shown in Display P-1.



Display P-1

The symbols <**cr**> also appear in Display P-1. These symbols indicate that the user must press the Carriage Return key (sometimes labeled Return, CR, or Enter).

The **A>** characters indicate that CP/M is ready to begin processing commands from the diskette in disk drive A.

You'll also see the caret or circumflex symbol (^) used in this guide. It indicates that you should press and hold down the key marked **CONTROL** or **CTRL** (**ALT** for IBM machines) and then type the following character. For example, ^**C** means that the Control key is to be pressed and held down while the **C** key is pressed, not that you should actually type the ^ before the **C**. Using the Control key is similar to using the Shift key; you hold it down while you type the character.

1 ABOUT CP/M

CP/M is an operating system for microcomputers. An operating system is a large program which controls the management of the computer's resources for you, the computer user. As your interface with the computer, CP/M supervises the movement of data through the various components of the computer, manages the creation, reading, writing, and deletion of files, provides you with reports on the status of files, and controls the loading and execution of other programs.

To better understand how CP/M works, we can draw an analogy between an operating system and a bank teller. When you go to the bank, there are a variety of jobs the teller can perform for you: deposit or withdraw funds, transfer funds from one account to another, inquire into the status or make adjustments to accounts, open or close accounts, et cetera. You have the teller perform these operations because you are not familiar with all the procedures and routines required to perform them according to bank standards. The teller is your *interface* with the bank.

CP/M is the "teller" for the microcomputer environment. You go to CP/M to "deposit" or "withdraw" data, transfer data from one file to another, inquire into the status of or make adjustments to files, create and erase files, and so on.

Like a bank teller, CP/M does not know which transactions you want performed until you specify them. When CP/M receives a command it recognizes, it will promptly go to work, but the command must be specific. For example, saying to a bank teller: "I want to transfer funds from one account to another," is not enough. You must supply the following information:

1. The amount of money to be transferred.
2. The number identifying the account from which funds are to be withdrawn.
3. The number identifying the account to which funds are to be deposited.

These details are the *parameters* of the transaction. Once you have supplied this information, the teller will execute the transaction accord-

ing to bank procedures; as the customer, you do not participate in the actual typing of account numbers and dollar amounts on the teller's terminal, nor do you open the cash drawer, count money, or issue receipts. The teller performs all the small duties associated with the particular transaction. When you issue commands to CP/M, you will often specify certain parameters. CP/M will then perform the command according to the standards of the microcomputer.

With CP/M handling all of the internal functions in a consistent and orderly manner, application programs don't have to be written to manage the routine functions of computer operation. They take advantage of CP/M's capacity to create files, control the printer, and so on. This also means that application programs can run on any computer that supports CP/M—so long as the application program is CP/M compatible. CP/M compatible means, among other things, that the organization of data within the program conforms to the way CP/M structures its files.

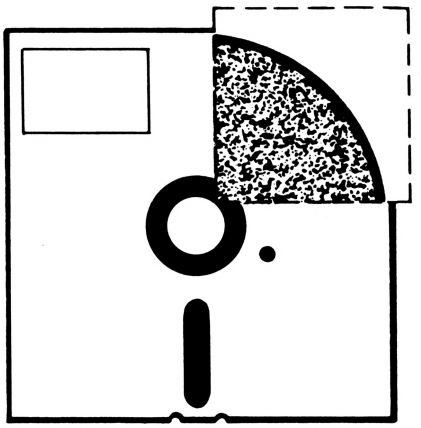
1.1 DATA STORAGE AND FILE STRUCTURE

The organization of data into file structures begins with the layout of the actual recording surface, or medium, of the floppy diskette.

The recording medium of a diskette is organized into *tracks* and *sectors*. Tracks form a series of concentric bands around the diskette; sectors are equal portions of those bands, measured in bytes—usually 128, 256, or 1024 (see Fig. 1-1). The configuration of tracks and sectors (called the format) of the diskettes you will be using depends on the requirements of your microcomputer system. Ask your computer vendor or consult the documentation that came with your system for the appropriate diskette format. Sometimes it is possible to buy preformatted diskettes. However, most CP/M releases include some kind of formatting program that is easy to use. Even if you buy preformatted diskettes, it's a good idea to format them yourself, if possible. Occasionally, during shipping or storage, diskettes may be exposed to conditions which may erase all or part of the format. You'll save time by making a practice of formatting each new diskette before you use it.

In order to organize data into files, CP/M superimposes a second pattern on this structure of tracks and sectors. CP/M arranges information in *records* of 128 bytes. If a diskette is formatted with 128-byte sectors, one record will directly correspond to one sector. If the sectors are 256 bytes, then there will be two records per sector; and 1024-byte sectors will contain eight records.

CP/M allocates and reclaims space on a diskette in segments called *allocation blocks*. An allocation block may consist of between 1 and 16 kilobytes (defined by the installer who implemented CP/M on a particular computer system).



Tracks form a series of concentric bands around the diskette and are numbered from 0 on, beginning at the outer edge.

Sectors are equal portions of a track, measured in bytes, and are numbered from 1 on.

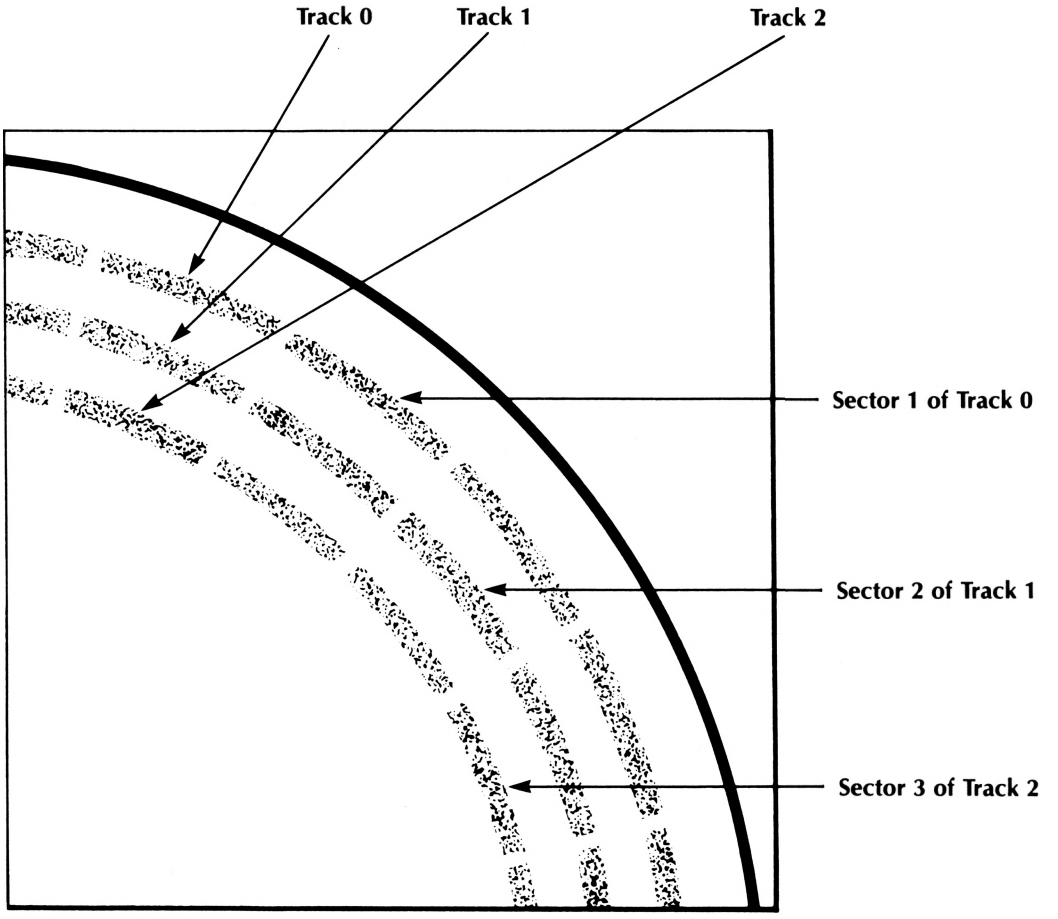


Fig. 1-1. Configuration of tracks and sectors on a diskette.

CP/M groups allocation blocks together to form an *extent*. Files are composed of one or more extents.

As you start to use CP/M more and more, you'll see that diskette and file statistics which concern memory space are always described in kilobytes (you'll see this represented as "6K" or "6KB") and records. (See the section on using the STAT command in Chapter Two, page 57.)

1.2 FILE NAMES AND EXTENSIONS

Any collection of related data which is stored as a separate unit is a file, and every CP/M file has a unique name. A file name (sometimes designated as *filename* or abbreviated *fn*) consists of 1 to 8 characters. Any characters, alphanumeric or symbolic (see "ASCII" in the Glossary), appearing on your keyboard may be used to create file names except:

```
[ ] * ? = : ; , . < >
```

and an empty space. These characters are reserved for use in CP/M command lines.

Files may be further identified with an *extension* or *type* (variously referred to as "filetype," "file extension," "file reference," or abbreviated ".ext" or ".typ"). An extension is optional and may consist of 1 to 3 characters (with the same restrictions as above), separated from the file name by a period, as in the examples below:

```
STAT.COM        5MAY81.REC
REPORT.TXT      MAILIST.SF
WEEK.001        WEEK.2
```

The name you give a file is up to you, but there are a few things you should consider. First, you cannot assign two files on one diskette the same name and extension. If you try this, you will either receive an error message or overwrite the first file with that name and extension, depending on which command you use to name the new file. Second, the name you choose should act as a mnemonic device, something that will help you remember what the file contains. Also, there are some conventions concerning file extensions. The list below shows extensions that are used by some of CP/M's commands (you'll see many of them explained in Chapter Two) and some popular application programs:

.ASM	Assembler Source File
.BAK	Backup File
.BAS	BASIC Source File
.COM	Transient Command
.HEX	Hexadecimal Code
.INT	Intermediate Code
.$$$	Temporary or Incorrectly Closed File (unusable)
.PRN	Printer Listing File
.REL	Relocatable Module
.SUB	Submit File
.SYM	Symbol File
.TXT	Text File
.OVR	Overlay File

1.3 DISKETTE CARE

To avoid damaging your diskettes and thereby losing all or a portion of the data they contain, follow these general precautions:

- Do not expose the diskette to magnetic fields, including magnetized tools and heavy electrical equipment. Exposure to even a relatively weak magnetic field, such as near a telephone, can destroy data stored on a diskette.
- Avoid physically damaging the diskette's recording surface by writing only on the diskette label, and only with a felt tip pen (don't use a ball-point pen or a pencil).
- When you're not using the diskette, keep it in its protective envelope and store it in an upright position. Replace an envelope whenever it becomes excessively worn or damaged.
- Do not expose a diskette to:
  —sunlight
  —liquids
  —smoke
  —hot or contaminating substances
- Never touch the recording surface or bend the diskette.

You may *write-protect* a diskette so data cannot be inadvertently written onto its recording medium. As shown in Fig. 1-2, a 5.25-inch diskette has a write-protect/enable notch on one edge. Placing a write-protect tab over this notch will prevent data from being written onto the diskette. These adhesive tabs are included in a box of diskettes. Fig. 1-3 shows a write-protected 5.25-inch diskette.

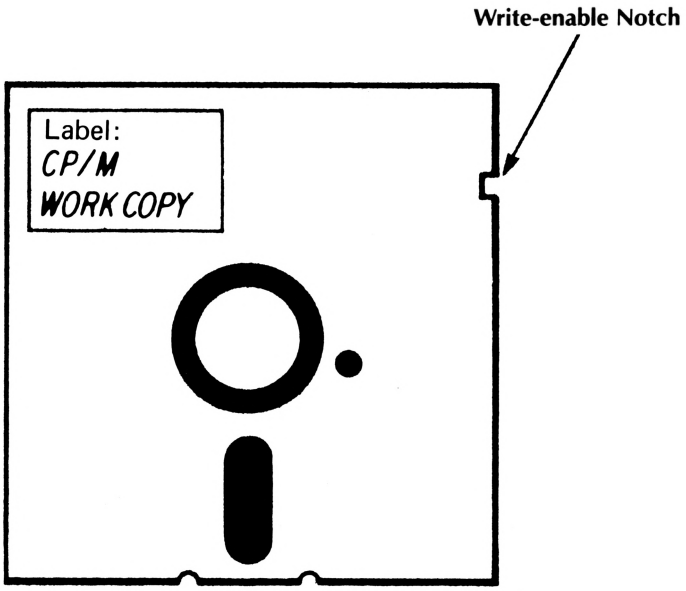


Fig. 1-2. Write-enabled 5.25-inch diskette.

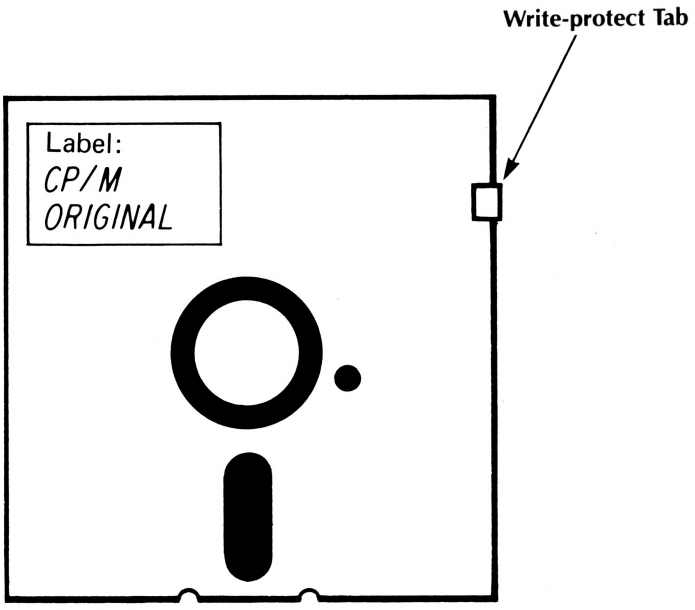


Fig. 1-3. Write-protected 5.25-inch diskette.

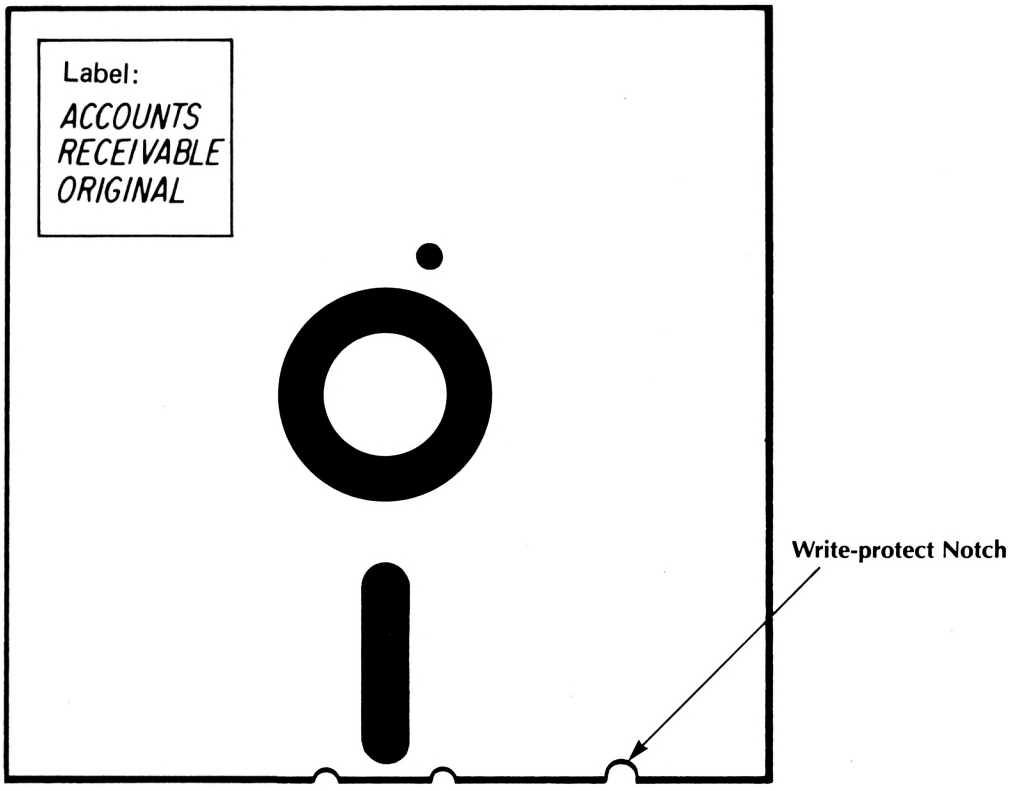


Fig. 1-4. Write-protected 8-inch diskette.

For an 8-inch diskette, it's the opposite: an *OPEN notch will PREVENT* data from being written on the diskette, while a *closed* notch will allow it to be written on. Fig. 1-4 shows a write-protected 8-inch diskette.

It's a good practice to write-protect every diskette containing software you have purchased; it guards against accidental corruptions of original copies.

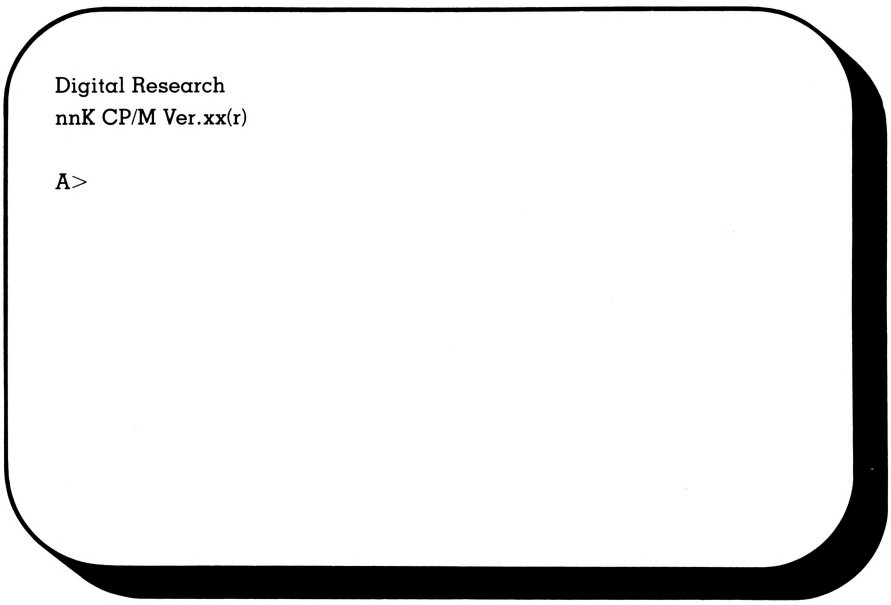
If you attempt to format or place data onto a diskette that is write-protected, an error message will appear on the console.

1.4 LOADING CP/M

Before you can begin executing CP/M on your computer, you must first load the operating system into the computer's memory. This function is also called *booting* or *bootstrapping* the system. In most instances all this really means is that you must turn on the power for the computer and the disk drives, place a diskette which contains CP/M into drive A, and then depress the Reset key (consult the User's Guide that came with your computer for more specific loading instructions).

When this happens, the computer reads a small program from the first sector of the first track on the diskette and places it in its memory. This occurs every time power is applied to the computer (called a *coldboot*). The small bootstrap program contained in Track 0, Sector 1 of the CP/M diskette instructs the computer to load a larger program (the CP/M operating system) into memory beginning at a predesignated location.

When the CP/M system has successfully booted, the screen of your computer, or terminal, will contain a message similar to the one shown below in Display 1-1.



Display 1-1

The nnK number is used to indicate the *size* of the CP/M system (K denotes kilobytes). This provides a guide to how large an application program you may run. (Software packages will often indicate "Requires a 52K CP/M," for example). Ver.xx is the particular version of CP/M (which will most likely be 2.2) and the (r) identifies the particular release of that version.

Always follow the instructions in the User's Guide that came with your computer to load a diskette. If you are loading your end-user distribution copy of CP/M, be sure it's write-protected (notch open for 8-inch, closed with a tab for 5.25-inch) before placing it in the disk drive.

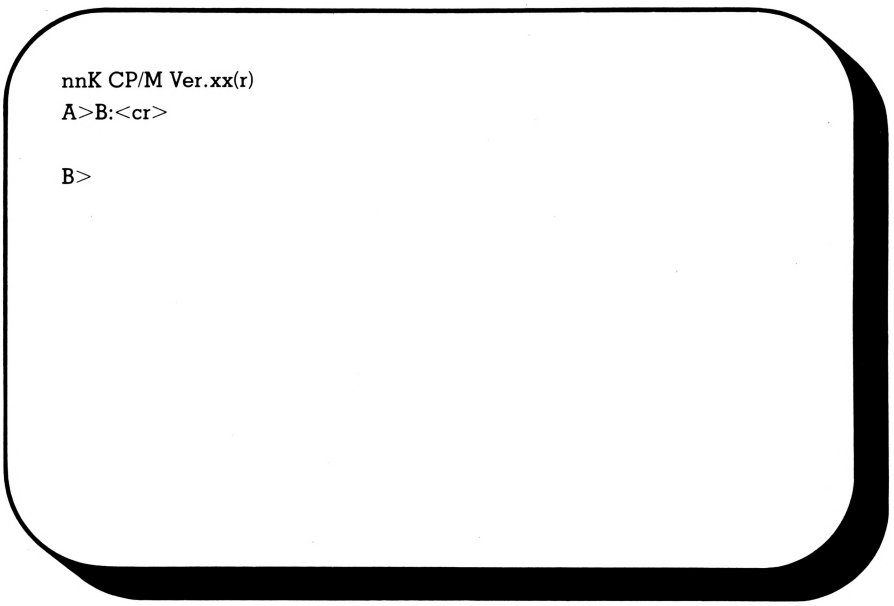
1.5 CP/M PROMPT AND DRIVE DESIGNATION

When CP/M has successfully loaded itself, along with the sign-on message you'll see A> displayed at the left of your screen. This is CP/M's prompt. It tells you two things:

1. CP/M is ready to begin accepting commands.
2. All commands will use drive A, unless another drive is specified.

The first item needs little explanation: when you type a valid command line in from the console, CP/M will execute it.

The second item is CP/M's default drive designation. This means that the CP/M system you have loaded into your computer came from the diskette in disk drive A, or the first disk drive attached to your computer. Drive A is logged in; from the CP/M system on drive A you can switch to any other drive. This is accomplished by simply typing the letter for the drive you wish to be switched to, followed by a colon and a carriage return (B: for example). When you have switched the currently logged-in disk drive, you'll see a new prompt, as shown in Display 1-2. (A CP/M compatible diskette must be placed in the drive before you issue B:.)



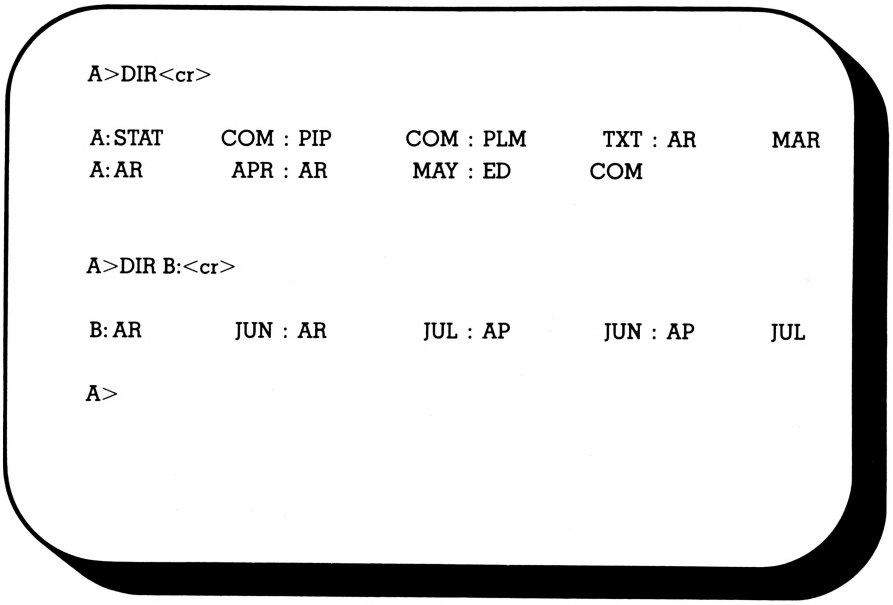
Display 1-2

1.6 GENERAL COMMAND STRUCTURE

You may issue a command to CP/M by typing a command line on your console after the CP/M prompt is displayed. A command line consists of the name of the command, the conditions or options associated with that command (*parameters*), and a carriage return. Some of the more frequently specified parameters are described below:

- **Disk Drive Designation** You may want to access, alter, or move a file that is not located on the currently logged disk. In this case, you would specify the required disk drive after typing the command. This is done by typing the drive designation (usually A, B, C, and D on a four-drive system), followed by a colon.

  Display 1-3 shows the difference between one command that does not specify a disk drive and one that does. The DIR command requests that a directory of all the files on a diskette be displayed. In the first command, no drive is specified; consequently, the directory consists of the files found on the default drive, drive A (as indicated by the A: to the left of each directory line). The second time the command is issued, drive B is specified.

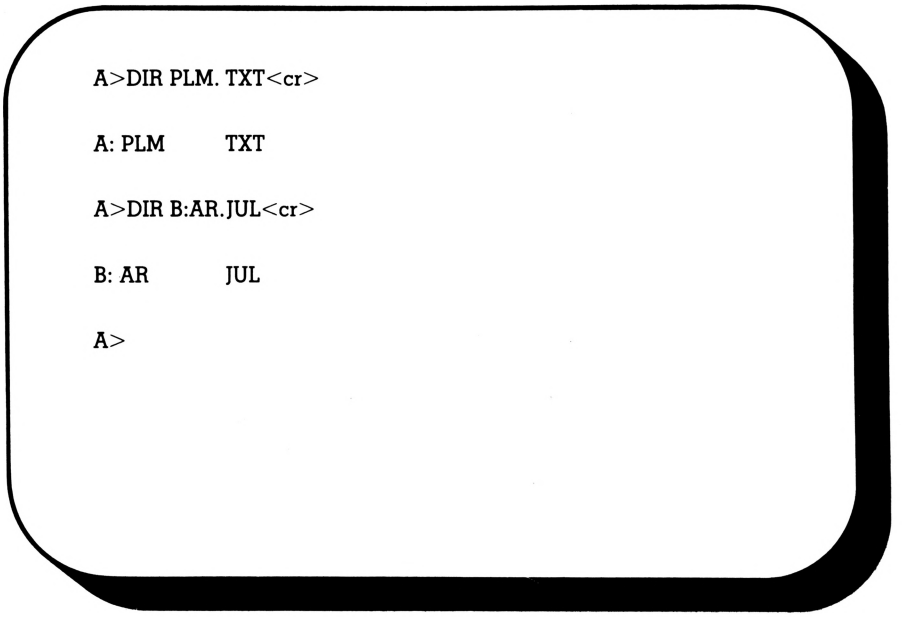


Display 1-3

- **File Name and Type** Usually, you will identify the file to be acted upon by the command. A file is identified in the command line by typing the file name, followed by a period (.) and the file extension. The file name may be preceded by a disk drive designation. The

command will then be focused on a certain file contained on the diskette within a certain disk drive.

Display 1-4 shows two command lines in which a file name has been identified.



Display 1-4

The first command line in Display 1-4 does not designate a disk drive, so CP/M assumes the default drive, drive A, and checks the directory of the disk for the file PLM.TXT as identified by the user. In the second command line, the user has designated disk drive B and identified the file AR.JUL so that the DIR command will check for that file in the directory of the disk currently in drive B.

- **Ambiguous and Unambiguous File References** A file identified in a command line is a *file reference* and may be of two types. The examples shown in Display 1-4 are *unambiguous*; PLM.TXT and AR.JUL are specific references. Each command line contains the unique name and extension of a single file.

  The second type of file reference is called *ambiguous* because the name of the file is not uniquely identified. An ambiguous file reference is used to define a group of files with some common elements in either the file name or extension (or both).

  The forms of both types of references are similar, but an ambiguous file reference makes use of the ? and * symbols to indicate that portion of a file name or extension which is common among a group of files. For example, where an unambiguous file reference would appear as:

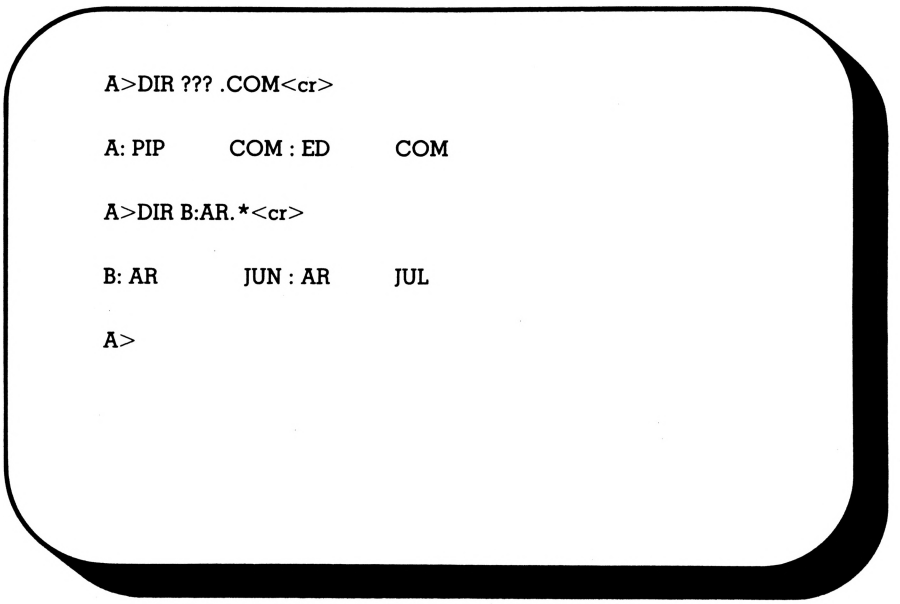
PLM.TXT

an ambiguous file reference could look like any of the following:

PLM.??? PLM.* ???.TXT *.TXT

Notice that the ? is used to substitute a single character in a file reference, and the * takes the place of an entire file name or extension. The ? may be placed anywhere within the ambiguous file reference. *.TXT is an abbreviated form of ???.TXT. Both references would cause the command line to act upon files on the diskette (in the designated drive) that have an extension of .TXT. The ???.TXT reference specifies that the files must have a name no longer than three characters. The *.TXT reference specifies that the command act on all files with an extension of .TXT regardless of the number of characters in the file name.

Display 1-5 shows two command lines that make use of ambiguous file references.



Display 1-5

The first command line in Display 1-5 contains no drive designation, so CP/M assumes drive A, and its ambiguous file reference requests the DIR command to list all the files which satisfy the requirement of having a 1-to-3-character (???) file name and an extension of .COM. Even though there is another file on the disk

with .COM as an extension in drive A (STAT.COM), the command was specifically requested to act only on those files with a 1-to-3-character name, so STAT.COM does not show up on the display.

The second command line designates a directory listing of the files on drive B which have a name of AR. The file reference is ambiguous because the * symbol was used as the file extension; therefore, all the files in the directory of drive B with the name AR, regardless of their extensions, are displayed.

An ambiguous file reference may be used to cause the command to act upon all the files on the specified disk drive. Such references may appear as:

```
*.*     *.???     ????????.*     ????????.???
```

(There are 8 ?'s because 8 characters are the upper limit for a file name; remember, the number of ?'s always represents the number of characters in the file name or extension.)

There are many other command parameters whose special uses are associated with particular commands. Those that we feel will be of use to the intended audience of this book are described in Chapter Two.

2 CP/M COMMANDS

CP/M commands are generally grouped into three categories: line editing, built-in, and transient. The line editing commands, issued as one letter Control keys, are described in Section 2.2. The built-in and transient commands are more powerful. They are employed to access and manipulate the information contained in disk files and to control system devices. These commands are described in Sections 2.3 and 2.4, respectively.

The CP/M error messages most commonly encountered when issuing commands, and the conditions which cause them to be displayed, are discussed in Section 2.5.

So that you'll have an idea of what's going on within the operating system when you issue a command, Section 2.1 provides an overview of CP/M's internal structure.

It's important that you have a basic understanding of a command before you use it, so please read through this chapter completely before proceeding. When you are ready to begin experimenting with the use of the commands, issue them only from the work copy you prepare following the instructions in Chapter Three.

2.1 CP/M'S INTERNAL STRUCTURE

CP/M consists of five distinct segments: the Console Command Processor (CCP), the Basic Disk Operating System (BDOS), the Basic Input/Output System (BIOS), the Transient Program Area (TPA), and the System Area. Figure 2-1 is a generalized representation of how these segments are arranged in your computer's memory. The numbers to the left of the boundaries represent examples of memory addresses (or locations) in hexadecimal notation.

Each segment has its area of responsibility:

The CCP is the user interface with CP/M. It accepts user commands typed on the console, executes the built-in commands, and, in turn, interfaces with BDOS when transient commands are specified.

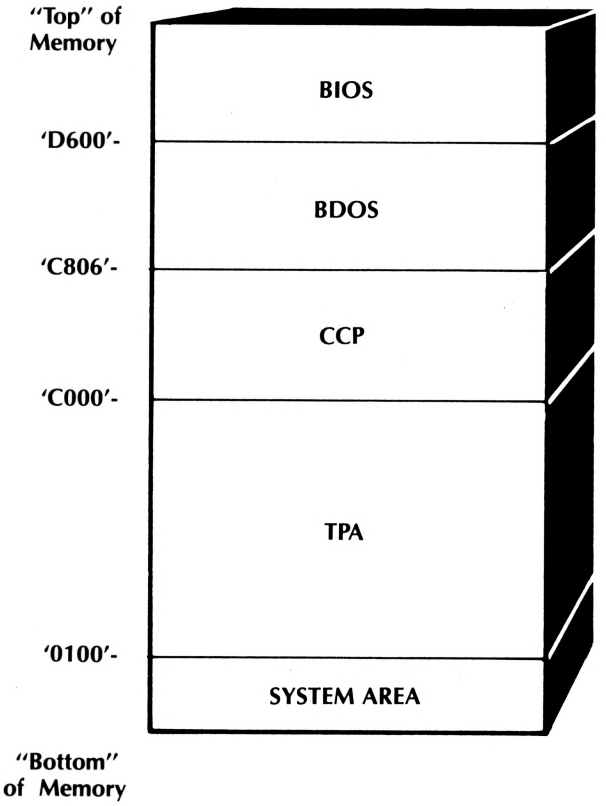


Fig. 2-1. CP/M's use of computer memory.

BDOS tasks include the actual reading and writing of information on a diskette, the allocation of space for records on the diskette, and directory searches. BDOS contains the logic which performs the internal functions of the system.

The BIOS contains the logic that controls the external functions of the system, those functions which are specific to the particular design of the system's hardware.

The TPA is a portion of the computer's memory that CP/M reserves so that programs not contained in the actual operating system may be loaded from diskette for execution.

The System Area occupies the lower portion of memory and contains special instructions and data areas unique to the particular implementation of CP/M.

When you, as the user, enter a command to CP/M through the console, CCP examines the characters to determine if they represent a valid command. If what you have typed is not a valid command, CCP sends an error message to your display. If the characters are a valid command, CCP processes it.

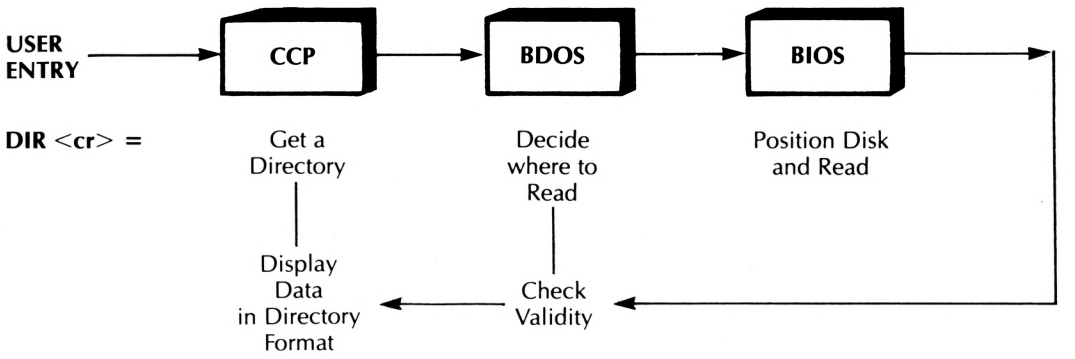


Fig. 2-2. Execution of built-in command.

If you have issued one of the transient commands, CCP passes information to BDOS that enables it to find where the .COM file for that command has been stored by BDOS. BDOS then informs BIOS where to find the file on the diskette. In effect, BDOS uses BIOS to perform the menial tasks associated with the operation of the disk drive, such as moving the disk drive head to the proper location on the diskette.

With BDOS managing the control of the disk drive through BIOS, the file containing the requested transient utility is placed in the TPA. When it is loaded into the TPA, CCP transfers control to the transient utility.

Figure 2-2 depicts, in a simplified manner, the processes involved in the execution of DIR, a built-in command. The CCP interprets the user entry DIR <cr> as a request for a file directory. It passes a request to BDOS to read the diskette for the information it will need for the directory. BDOS, because it is in charge of system level disk operations, determines where it will direct BIOS to read from the diskette. When BIOS has read from the specified location(s), it passes information to BDOS for validity checking. If the data are valid according to its instructions to BIOS, BDOS passes the information on to CCP, which causes it to be displayed in the appropriate format.

The execution of a transient command is depicted in Fig. 2-3. The CCP interprets the user entry of PIP as a request for the .COM file which contains the transient command PIP. The CCP then instructs BDOS to open and read the PIP.COM file on the diskette. BDOS directs the BIOS to read the necessary sectors to locate the file on the diskette and load it into the TPA. After BDOS has checked the validity of the data read from the given location, it moves the contents from the files into the Transient Program Area of the computer's memory (starting at 0100 in this example). Once PIP.COM has been loaded into the TPA, it begins the

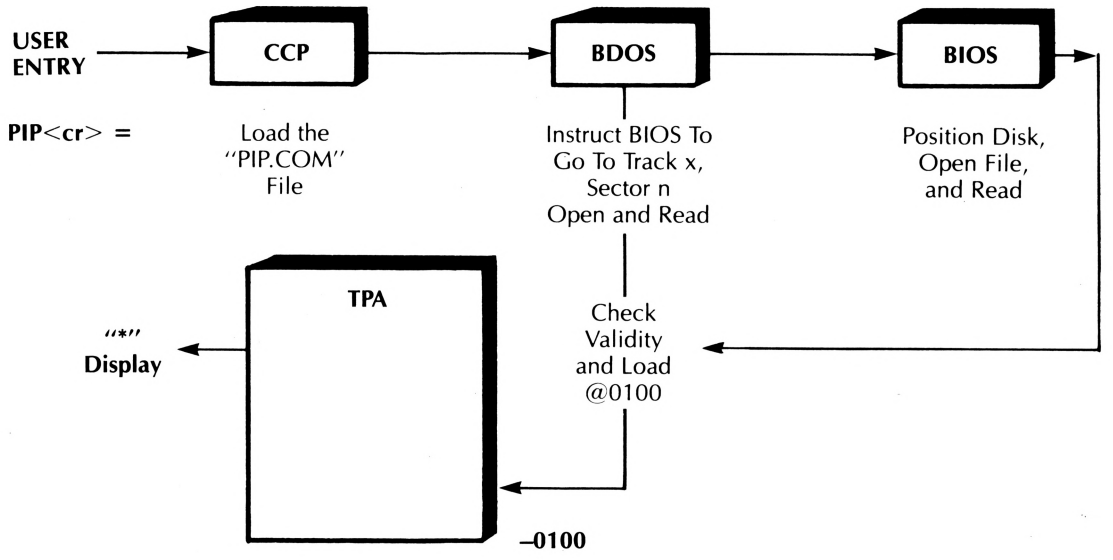


Fig. 2-3. Execution of transient command.

execution of the user-issued command (this is represented by PIP displaying its "*" prompt). When the user indicates that PIP will no longer be required, PIP returns control to CCP.

2.2 LINE EDITING COMMANDS

One set of CP/M commands allows you to correct errors or make revisions while typing command lines. We have grouped these commands according to their functions in Table 2-1. The ^ symbol indicates that the CTRL or Control key on your console should be held down or typed simultaneously with the character that follows it.

Some line editing command functions are duplicated by certain keys on your console. For example, ending and entering a command may also be achieved by typing the Carriage Return, Enter, or Line Feed keys. In many cases, you may save time by typing a single key instead of the two keys that are required for a line editing command.

The rest of this section contains detailed descriptions of CP/M's line editing commands.

Command: **^C Warmboot/Start**

This command executes a warm start, returning CP/M to the computer's memory. ^C will be used primarily to let CP/M know you have

Table 2-1
Line Editing Command Functions

LINE EDITING COMMANDS	FUNCTION
^C	CP/M Warm Boot
^J ^M	End and Enter the Command Line
^H	Cancel the Last Character in the Command Line
^U ^X	Cancel Entire Command Line
^E ^Q ^R ^S	Commands for Controlling the Display
^P	Printer Output Control

placed a new diskette in one of the disk drives. If CP/M does not know you have changed diskettes, it can give you inaccurate information, fail to find the file you want, or destroy the Directory of the new diskette.

^C can generally be used to escape from a program or utility if it is entered when console input is expected. Note that ^C must be the first character of an input line to produce these effects.

Command: **^E Position Cursor at Beginning of Next Line**

When you are entering a long command line, or when console input is required that takes up more space than is available on one line of the console display, entering this command will position the cursor at the beginning of the next line. Then, when you're ready to depress the carriage return, CP/M will read the complete entry as a single command, although it reads as several lines on the display. ^E causes a carriage

return that CP/M will not acknowledge as the entry of a command line.

Command: **^H Erase Character from the Screen**

This command deletes and erases the last character you typed. ^H performs the same function as the Delete (DEL) and Backspace (BS) keys.

Command: **^J or ^M Carriage Return, Line Feed**

This ends the command line and moves the cursor to the beginning of the next line. The same function is provided by a Line Feed or Carriage Return key. When ^J or ^M are entered, CP/M will execute the line of type as though the user had depressed the <cr> key.

Command: **^P LST: Device Output On/Off**

When ^P is typed, all output going to the screen will also be sent to the LST: device—usually a printer. To activate the printer, type ^P once; to stop printing, type it again. Typing ^C will also stop printing invoked by the ^P command and return control to CP/M.

Command: **^Q Continue Output to the Screen**

^Q will resume scrolling (output going to the screen) after it has been halted by a ^S command.

Command: **^R Retype Command Line**

When you have used the echoed Delete or Rubout key to correct typing errors, your command line can be difficult to read. Issuing a ^R command will show you what the command line really looks like by spacing down one line, retyping it, and positioning the cursor after the last character, from which you may continue typing.

If your version of CP/M supports Delete, Backspace, and Rubout keys which are not echoed on the console display, you will have

little use for this command. ^R displays a # symbol to indicate where it was used.

Command: **^S Stop Output to the Screen**

This is where ^S comes in: it pauses the display of information on the screen. Depending on the version of your copy of CP/M, you can continue screen output by entering ^S again, ^Q, or any other Control key. Some computers have a No Scroll key that will pause the display the first time it is depressed and then resume it when depressed again.

Command: **^U Cancel but Do Not Erase Command Line**

When you've typed in a command line and made an error that you can see, entering the ^U command will cancel it so that it will not be executed but it won't erase it from the display. Even though CP/M will ignore the line typed before the ^U was entered, it will remain on the display. In this way you can copy that part of the command line that was not in error. ^U displays a # symbol to show where it was issued.

Command: **^X Cancel and Erase Command Line**

If you have made an error in entering a command line, typing a ^X will erase it from the screen and return you to the prompt. It will not work if you have already entered a Carriage Return.

2.3 BUILT-IN COMMANDS

Built-in commands are those which permanently reside within CP/M; they are loaded into the computer's memory with the operating system. Because these commands are contained in the system, they will not be listed in a file directory. Built-in commands are available for use any time CP/M is active.

The built-in commands are:

DIR	Displays a directory of the files on a diskette.
ERA	Erases a file or files from a diskette.
REN	Renames a diskette file.
SAVE	Saves a specified amount of data from the TPA in a diskette file.
TYPE	Displays the contents of a file.
USER	Accesses a specified user area.

This section contains a comprehensive description of each CP/M built-in command. The descriptions include a narrative explanation of the command's function, use, parameter options, and one or more examples.

DIR

Basic Structure: DIR<cr>

The DIR command causes a directory of disk contents to be displayed on the console device. This directory consists of the name and type of each file, and is organized on the screen in four columns separated by colons, as shown in Display 2-1.

```
A>DIR<cr>

A: ED       COM : SYSGEN   COM : COSTS   JAN : INV     DAT
A: STAT     COM : SUBMIT   COM : COSTS   FEB : COSTS   MAR
A: PIP      COM : ASM      COM
```

Display 2-1

Notice that the file names are separated from their extensions by a number of spaces. These spaces are not a part of the name; for example, what appears as ED COM is actually the file ED.COM. The DIR command uses this method of organization for convenience and to enhance readability. CP/M provides enough space for a file name with the maximum of 8 characters. The A: indicates that these files are contained on the disk in drive A.

Examples:

DIR<cr>	Display a directory of the files present on the currently logged disk drive.
DIR B:<cr>	Display a directory of the files present on disk drive B.
DIR B: *.TXT<cr>	Display a directory of all the files present on disk drive B with the file type .TXT. The file type may also be ambiguous (COSTS.*).

Display 2-2 shows what the execution of the last two examples might look like on the screen.

```
A>DIR B:<cr>

B: REPORT   TXT : COMPANY  TXT : STAT       COM : MAILIST  83
B: PIP      COM : RECORD   PRN : PRELIM     TXT : SYSGEN COM
B: USER     TXT

A>DIR B:*.TXT<cr>

B: REPORT  TXT : COMPANY  TXT : PRELIM    TXT : USER     TXT

A>
```

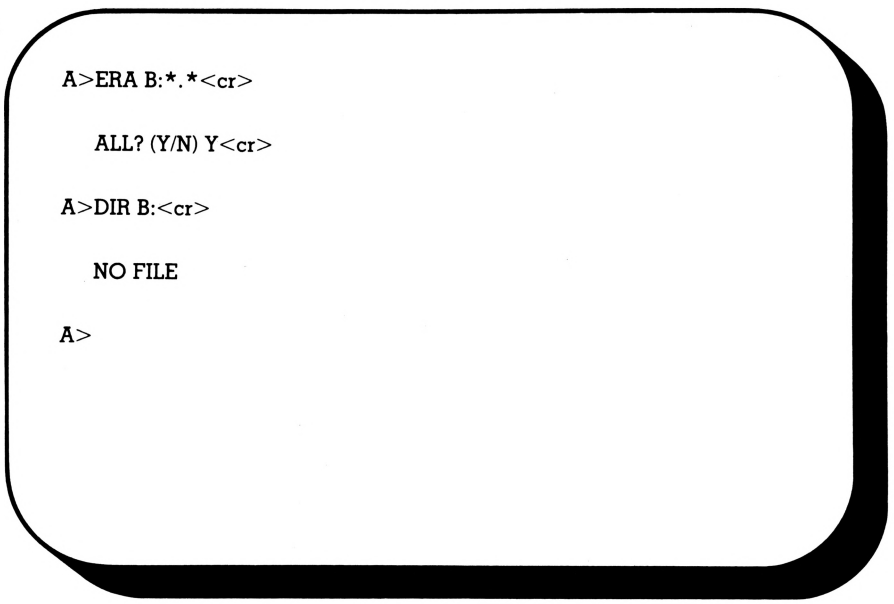
Display 2-2

ERA

Basic Structure: ERA name.ext<cr>

The ERA command is used to erase files from a disk. Ambiguous file names and extensions may be designated. A disk drive may also be designated, but if omitted the command defaults to the currently logged drive. If you attempt to erase a file from a diskette that is write-protected, the error message "Bdos Err On x: R/O" (where x: is the drive specified in the command) is displayed.

A word of caution: You should be very careful whenever you use the ERA command. If you inadvertently erase a file that you must keep, it's too late; your file is gone. You cannot recover a file once you have erased it. Be especially careful when using ambiguous file references in an ERA command. Make sure that no file you intend to keep falls in the category of the ambiguous reference. Display 2-3 shows what happens when the ambiguous file reference *.* is used in an ERA command. A safe procedure is to verify diskette contents using the DIR command before using ERA.



Display 2-3

When you command CP/M to erase all the files on a diskette, CP/M will ask you if you are sure this is what you want to do. You reply by typing either Y for yes or N for no. In the above display, the user

approves the erasure, then asks for a directory of the diskette just erased; no files remain on the diskette.

Examples:

ERA OBSOLETE.TXT<cr>

Erase the file OBSOLETE.TXT from the currently logged drive.

REN

Basic Structure: **REN newname.ext=oldname.ext<cr>**

The REN command renames a specified file. After typing REN, you type the new name you want the file to have, including its extension, then type an equals symbol (=), followed by the old file name and extension. Both the new and old names must be unambiguous. A disk drive may be specified after REN is typed, as shown in the second example below. The same drive may be specified after the equals symbol, but it is not necessary to do so. If a different drive is specified, an error message will be displayed.

Examples:

REN FOUND.DAT=LOST.DAT<cr>

Assign the new name FOUND.DAT to the file previously named LOST.DAT on the currently logged disk.

REN C:FOUND.DAT=LOST.DAT<cr>

Same as above, except that the name is to be assigned to a file present on drive C.

REN C:FOUND.DAT=C:LOST.DAT<cr>

Same as above.

SAVE

Basic Structure: **SAVE nn name.ext<cr>**

This command saves a specified portion of the data in the TPA by storing it in a diskette file. The command line specifies the amount of data to be saved (nn), and the name to be assigned to the file (name.ext). The disk drive may also be specified by the user.

As a beginner, you will probably use the SAVE command for only one purpose—to activate a new user area. Our description of the USER command contains some basic information about SAVE and an example of its use with both the USER and DDT commands.

SAVE is most often employed during an assembly language programming session in conjunction with the DDT program. If you would like more information about assembly language programming using CP/M, see the Digital Research manual, "CP/M Assembler (ASM) User's Guide."

Example:

SAVE 15 B:HOUSE.DOC<cr>

Save 15 pages of the TPA, on disk drive B, in a file named HOUSE.DOC.
Note: A page is a 256-byte block of memory, or the equivalent of two standard CP/M records.

TYPE

Basic Structure: TYPE name.ext<cr>

The TYPE command causes the contents of a file to be displayed on the console device. The file name and extension must be unambiguous. The disk drive may be specified. It is important to note that the file you wish to be displayed must consist of "printable" characters. That is, the information contained in the file must be written in ASCII code; if it isn't, only an unintelligible output of characters will be displayed. Files with the extensions .ASM, .BAK, .BAS, .DAT, .DOC, .HEX, and .TXT can generally be displayed using the TYPE command; files that have .COM or .OBJ extensions cannot.

Most files contain more information than will fit on the screen at one time. The information will be scrolled upward until the end of the file is reached, unless the command is issued to pause the output. This is done by typing ^S. When ^S is typed again, output will resume. To stop the typing and return to CP/M's A> prompt, just enter any character while output is taking place.

The ^P command may also come in handy if you want the contents of the file to be printed at the same time they are being displayed. Just type ^P after you have typed the name and extension of the file to be displayed, then press the Carriage Return or Enter key to issue the command. (For more information about the ^P and ^S commands see Section 2.2 "Line Editing Commands.")

Example:

TYPE B:LEDGER.BAS ^P<cr>

Display, from drive B, the file LEDGER.BAS, printing the file at the same time.

USER

Basic Structure: USER n<cr>

The USER command is issued to enter user area n, where n is a number from 0 to 15, inclusive.

Each CP/M file is assigned a user number from 0 to 15. A user area is an imaginary region made up of the files whose numbers correspond to the area. This means that once a number has been assigned to a file, you must issue the USER command to reference that file.

User areas can be activated when two or more people share a diskette, so that each person's files are contained in a separate logical area of that diskette. This prevents one user from accidently altering or erasing files which belong to someone else. Files which pertain to different projects, or deal with different subject matters may also be separated by assigning them to individual user areas.

```
A>DIR<cr>

A: ED       COM : SYSGEN   COM : COSTS   JAN : INV      DAT
A: STAT     COM : SUBMIT   COM : COSTS   FEB : COSTS    MAR
A: PIP      COM : ASM      COM : DDT     COM

A>USER 5<cr>

A>DIR<cr>

NO FILE

A>USER 0<cr>
```

Display 2-4

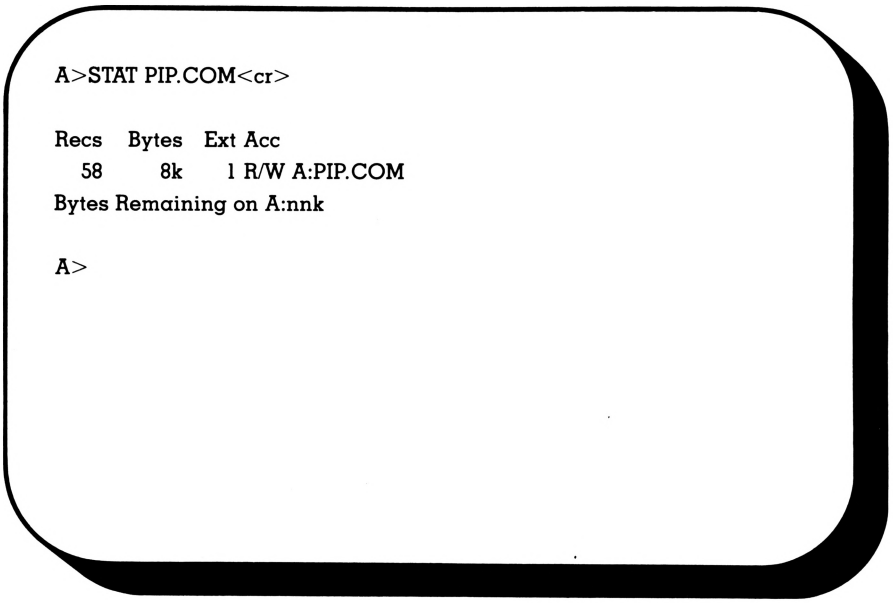
When CP/M is coldbooted you are in user area 0. All of the files that you see listed by the DIR command after you have booted the system have been assigned to user area 0. You may enter another user area at any time, but it will not become active until a file has been assigned to it. In Display 2-4 we see that many files may be referenced from user area 0. But when user area 5 is entered and a DIR command is issued, the message NO FILE is displayed by CP/M because no files could be found with number 5 assigned to them; user area 5 has not been activated. USER 0<cr> is then issued to return to user area 0.

Four commands are used to activate a user area: STAT, the transient command that shows file size; DDT, a transient command; SAVE, a built-in command; and PIP, the transient command used to transfer files. The remainder of this section describes the steps involved in activating a new user area with these commands. You may want to read more about the transient commands, PIP and STAT in particular, before you proceed (Section 2.4 "Transient Commands").

The first file you need to assign to a user area is PIP.COM, the file for the PIP command. With PIP, you'll be able to place other files in the newly activated user area.

1. Determining File Size

The first step in activating a user area is to determine the size of the PIP.COM file. Use the STAT command to do this, as shown in Display 2-5.



Display 2-5

The command line STAT PIP.COM requests STAT to display the size of the PIP.COM file. What we're looking for among all the information displayed by STAT's response is the number of records in the PIP.COM file, which STAT shows under the Recs heading. Our example shows that PIP.COM is made up of 58 records. Write down the number of records (you'll be using this information later).

2. Placing the File in the TPA and Entering Another User Area

The command lines involved in the second step of activating a user area are shown in Display 2-6.

```
A>DDT PIP.COM<cr>

DDT VERS 2.x
NEXT  PC
1E00  0100
-^C

A>USER 5<cr>

A>
```

Display 2-6

Typing DDT PIP.COM <cr> directs DDT to place the PIP.COM file in the TPA. When DDT does this, it displays under the NEXT PC heading, the beginning and ending of the PIP.COM file within the TPA. PC stands for *P*rogram Counter and NEXT is the next available memory location. The (-) character is DDT's prompt. Notice that our example shows the user typing ^C to return to CP/M. This leaves PIP.COM in the TPA and permits entry into user area 5 by typing the command USER 5<cr>.

3. Saving the Contents of the TPA

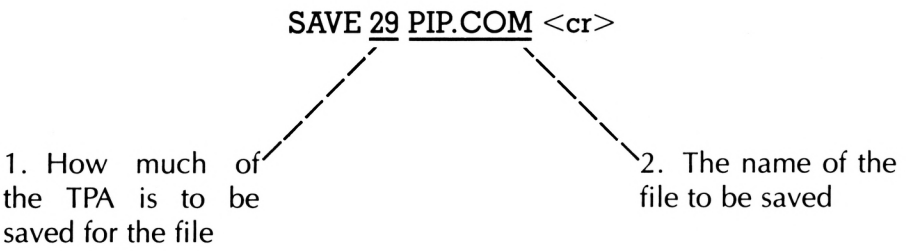
Entry into user area 5 is confirmed by the return of CP/M's A> prompt. It is as yet unactivated, and even though all the built-in commands are available, they are of little use because user area 5 contains no files. This is where SAVE comes in; with the SAVE command, we can take PIP.COM from the TPA and save it in a file

```
A>SAVE 29 PIP.COM<cr>

A>DIR
A: PIP      COM

A>
```

Display 2-7

which will be assigned to user area 5. Display 2-7 shows this third step in activating a user area.

When the SAVE command is used, two elements of the command line must be specified by the user:



The second item is obvious in this case, since PIP.COM was placed in the TPA. The first item is not so obvious. The SAVE command handles TPA memory in blocks of 256 bytes. Each block is called a *page* of memory. There are various ways of figuring out how many pages of memory a file occupies. An easy and reliable method is to divide by two the number of records obtained in step 1. The formula for our example would look like this:

$$58 \div 2 = 29 \text{ (pages to be saved)}$$

In step 1 you used the STAT command to determine the size of the PIP.COM file and recorded the number of records (58 in our example). Remember that all CP/M records are 128-byte units, so *two records equal one page* (2 x 128 = 256). If you divide the number of records by two, you get the the number of pages occupied by PIP.COM. If there are an odd number of records in a file, add one to that number and then divide by two.

Because the SAVE command in Display 2-7 was issued from user area 5, CP/M assigned that number (5) to the PIP.COM file.

Every user area shares the use of CP/M's built-in commands, but the commands will only affect the user area from which they are issued. In Display 2-7, when the DIR command was entered from user area 5, it displayed the only file assigned to user area 5—PIP.COM. There are now two PIP.COM files on the diskette, one assigned to user area 0, and one for user area 5.

Bringing Files into a User Area

Now that PIP is in user area 5, it can be used to bring in files from other user areas (usually 0, since it contains all the system files). PIP has a special parameter that allows the transfer of files from one user area to another. A command line using this parameter is shown in Display 2-8.

The first command line in Display 2-8 causes PIP to move a copy of the STAT.COM file from user area 0 into the currently active user area

```
A>PIP STAT.COM=STAT.COM[60]<cr>

A>DIR<cr>
A: PIP      COM : STAT      COM

A>STAT USR:<cr>

Active User : 5
Active Files: 0 5

A>
```

Display 2-8

(5). The "G0" characters contained in the square brackets specifically instruct PIP to get STAT.COM from user area 0; these characters represent the *user area parameter* of the PIP command line (any number from 0 to 15 may follow the G).

NOTE: Although PIP is more completely described in Section 2.4, it is important for you to know that this parameter only allows PIP to bring in files from another user area; *it will not permit you to transfer files from the currently active user area to another user area.*

Finding Active User Areas

If you should forget which is the currently active user area, the STAT command can assist you. A special form of a STAT command line will not only display the current user area, but it will also display all active user areas on the diskette. This is shown in the second command line in Display 2-8. In response to STAT USR:<cr> STAT shows that the Active User area is 5, and that there are files contained in areas 0 and 5 (Active Files). There is no way of finding out which files are contained in user area 0 unless you enter it.

Erasing Files from Other User Areas

Use of the ERA command is modified when several user areas are active on a diskette. The command line ERA *.* will erase only the files assigned to the currently active user area. To erase the entire diskette, you must enter each active user area in dividually and issue the same ERA command.

2.4 TRANSIENT COMMANDS

Transient commands, unlike the built-in commands, are separate programs contained in files on the CP/M diskette. They will appear in a file directory as .COM type files. To issue a transient command, just type the command name after the A> prompt. This will cause CP/M to locate the .COM file on the diskette and load it into the Transient Program Area (TPA).

Transient commands are set up as independent files because the programs (also known as *utilities*) which control them are too large to remain constantly in the computer's memory. One advantage to this is that you can create a CP/M diskette onto which you copy only the transient commands you expect to use most frequently, such as:

ED	Enables the user to create, examine, and alter CP/M files.

PIP	Transfers files from one diskette or device to another.
STAT	Displays the amount of diskette space occupied by a file or group of files and specifies system device assignments.
SYSGEN	Transfers a copy of the CP/M operating system to another diskette.

Our descriptions of these commands contain those aspects of their use which we feel will be of most interest to the beginning user. Error messages follow each command description.

The other transient commands, which are more generally used by programmers and experienced CP/M users, are not as fully described in this book. These commands are:

ASM	CP/M's assembly program.
DDT	Finds and corrects errors in an assembly language program.
LOAD	Converts an assembled program file into a COM type file so that it may be executed.
MOVCPM	Creates an image of the operating system which may then be generated to manage a specified amount of memory.
SUBMIT	Enables a series of CP/M commands to be contained in a file for sequential execution.
XSUB	Allows a Submit (.SUB) file to supply console input (as well as commands) to programs.

ED

CP/M's EDitor is used to create, examine, and alter files.

This description of the ED utility is meant to provide you with a knowledge of ED's operation and commands before you actually begin to use the editor. Once you are familiar with ED and have read Chapter Two completely, use the sample session in Chapter Five for hands-on experience with CP/M's commands. This description will also serve you later as a reference source.

Using ED to create and alter program source files (which are, in turn, submitted to an assembler or compiled for use within another language such as CBASIC), is not discussed in this guide. Our examples, which show the creation and alteration of a .TXT file, are followed by complete descriptions of ED's commands.

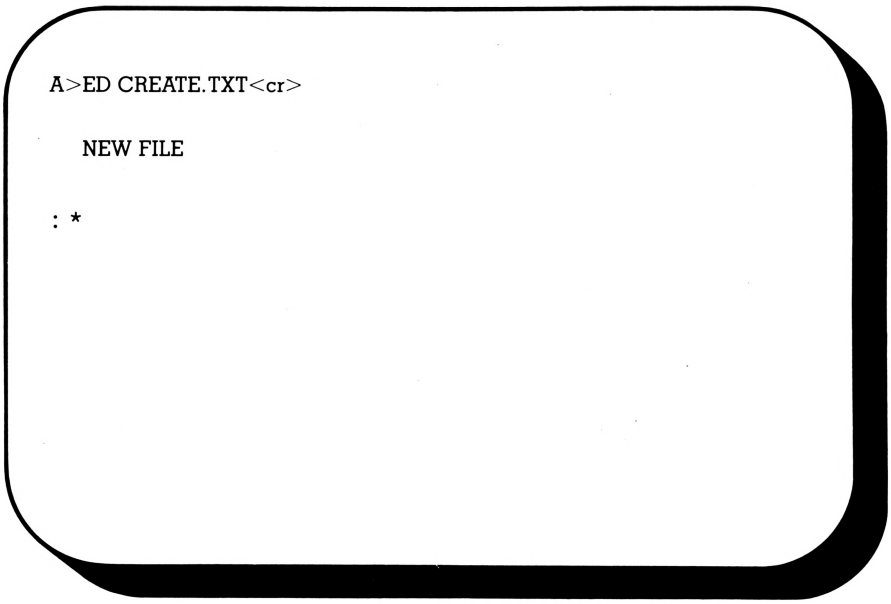
As you have seen, a .TXT file type designates a file made up of text: letters, common punctuation marks, and numerals. ED is well

equipped to handle small to medium size .TXT files, but if you intend to create large document files your task would be made much easier by using a word processor program. Many such programs designed to run under CP/M are available, and your vendor can probably recommend several.

Before we go any further, you should become aquainted with several elements of ED's operation.

- **The ED Prompt** When ED has been invoked, it *prompts* you to let you know that it is ready to accept your commands by displaying a colon (:) and an asterisk (*), separated by a space. ED is initiated by typing ED in a command line, followed by the name of the file you wish to create or alter (source file).
- **Text Buffer** When you create a new file or alter an existing one, your typed entries are not immediately written in a file on the diskette. ED stores your typed input in a portion of the computer's memory that it reserves. This portion of memory is called the text buffer, and your commands to ED and text entries affect only the contents of this buffer.

  ED assigns you line numbers for the text to be entered into the buffer, separating them when you type a carriage return. This is why ED is sometimes referred to as a line-oriented editor; you move through the buffer in increments of characters or lines. ED will not display the contents of its buffer as it will appear on the printed



Display 2-9

page. Your position within the buffer is directed through special commands.

- **ED's Character Pointer** When you move through the characters or lines of characters within the text buffer, your location is referenced by the Character Pointer (CP). Any commands you issue to alter or insert text will be effective relative to the position of the CP. It is important for you to remember that the CP will always be located either *ahead of* the *first* character within the *first* line, *behind* the *last* character of the *last* line, or *between* two characters. The CP remains located after the last character you insert until you issue a command to move it. The ED commands for positioning the CP and revealing its location are described in the section, "CP Control and Text Display" (page 41).
- **Source, Temporary, Backup, and Library Files** The operation of ED involves four types of files whose functions are denoted by their extensions:

  .TXT This is the source file which contains the latest edited version of a document and is the basis for the next version. The source file, identified in the ED invocation command (see Display 2-9), is assigned this extension by the user.

  .$$$ This extension designates a temporary file which ED creates in order to save the contents of the text buffer (the latest version of the text in your file).

  When you are creating a new file, ending your editing session causes the temporary file to be assigned the extension you named in the invocation command line. If you are altering an existing file (source), then the new version of the source file is moved into this .$$$ file through the text buffer, and its extension is renamed to that of the original file (.TXT).

  .BAK ED will always save the text of your source file as it was before the last edition in a .BAK file. This file is made at the end of the edit session when ED renames the original version of the source file. If you are using ED to create a new file, this BAcKup copy will be empty until you edit the first version.

  .LIB .LIB stands for Library, files you create to be read from so that text which is common and needs little or no revising can be pulled into the text buffer for the file you are currently editing. This type of file is used with the Rfile, X, and R commands described on page 39.

ED's routing of text through the source file, text buffer, and temporary file is depicted in Fig. 2-4.

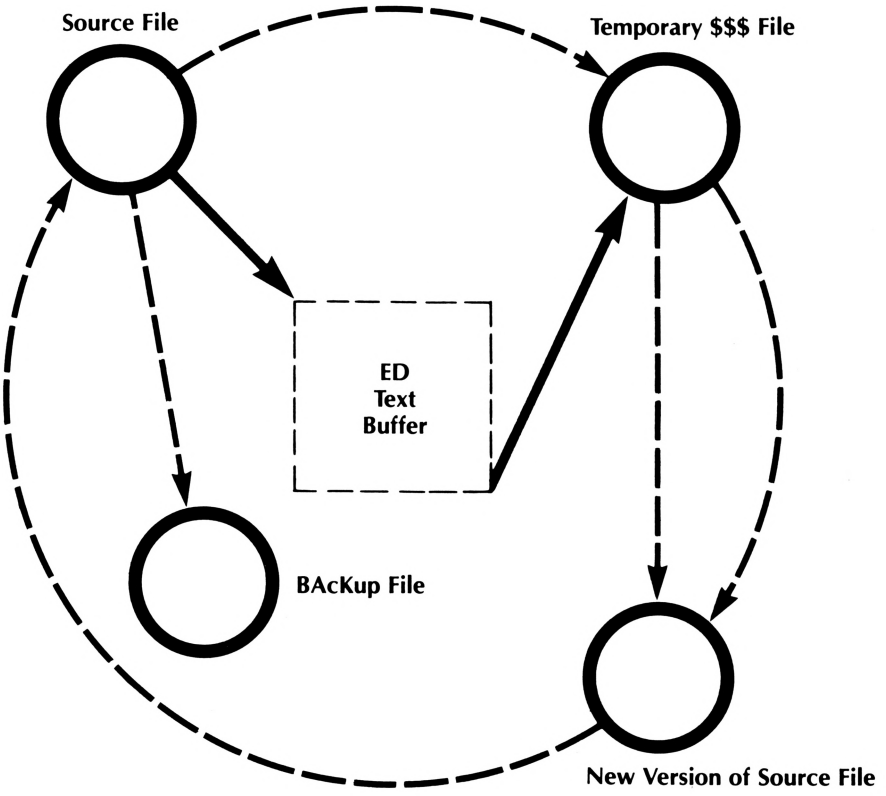
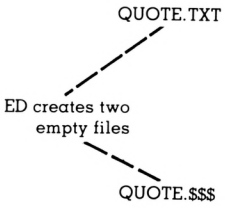


Fig. 2-4. Text transfer routing through ED.

A representation of ED's operation during the creation of a file is shown below. Complete descriptions of ED's commands follow.

```
A> ED QUOTE.TXT<cr>

NEW FILE
 : *I<cr>
1: A FLEA AND A FLY IN A FLUE<cr>
2: WERE IMPRISONED, SO WHAT COULD THEY DO?<cr>
3: SAID THE FLEA, "LET US FLY."<cr>
4: SAID THE FLY, "LET US FLEE."<cr>
5: SO THEY FLEW THROUGH A FLAW IN THE FLUE.<cr>
6: ^Z
```



TEXT BUFFER

A FLEA AND A FLY IN A FLUE
WERE IMPRISONED, SO WHAT COULD THEY DO?
SAID THE FLEA, "LET US FLY."
SAID THE FLY, "LET US FLEE."
SO THEY FLEW THROUGH A FLAW IN THE FLUE.

User-Entered Text Is Placed In The Text Buffer

Because the file QUOTE.TXT did not previously exist on the diskette, ED displays NEW FILE and creates two empty files. The first file ED creates is called QUOTE.TXT (as named in the command line); the second is called QUOTE.$$$. When ED assigns the extension $$$ to a file, it indicates that the file is temporary.

The file is temporary because ED will rename it later, at the end of the ED session. ED does not immediately write what has been typed into a file on the diskette. Instead, it reserves a portion of the computer's memory (called a *text buffer*) where it stores what the user is typing or *inserting* as text for the file. As shown above, the ED command for inserting text into the buffer is I, typing a ^Z takes ED out of the insert mode (line number 6).

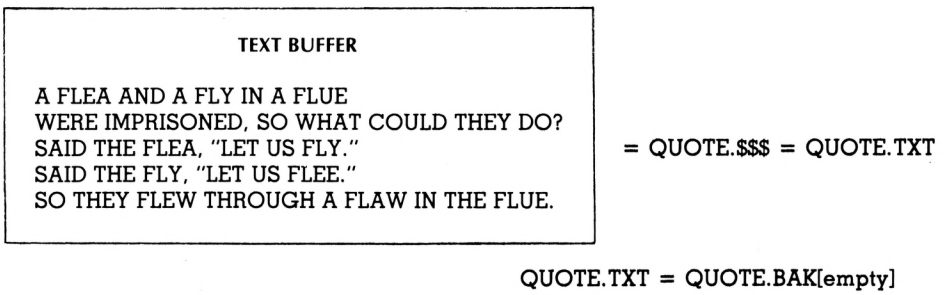
When the ED session is completed, the characters inserted in the text buffer are moved into the file called QUOTE.$$$, and this file is renamed QUOTE.TXT. The first file ED created is renamed QUOTE.BAK.

The .BAK extension indicates that the file is a backup copy. When you create a file, this .BAK copy is empty. Whenever you make revisions to a file, its original form will be assigned a .BAK extension and the new version will take the original's name.

In this way you are always assured of having both the current and the old version of your file. When you edit an existing file, that file is called the *source file* because it is the source of later versions.

The edit session in our example is ended with the E command. This causes ED to perform the transfer and renaming operations described above.

```
4: SAID THE FLY, "LET US FLEE."<cr>
5: SO THEY FLEW THROUGH A FLAW IN THE FLUE.<cr>
6: ^Z
 : * E<cr>

A>
```



Now that QUOTE.TXT has been created and stored, it may be used as the source file for a revised version. The examples below show the word SO being deleted from line 2.

```
A> ED QUOTE.TXT<cr>
```

Invoke ED to edit source file.

```
: * #A<cr>
1: *
```

Transfer all lines from the source file to the text buffer.

```
1: *2:<cr>
2: *
```

Position the CP at the beginning of the second line (line 2).

```
2: *F,^ZT<cr>
SO WHAT COULD THEY DO?
2: *
```

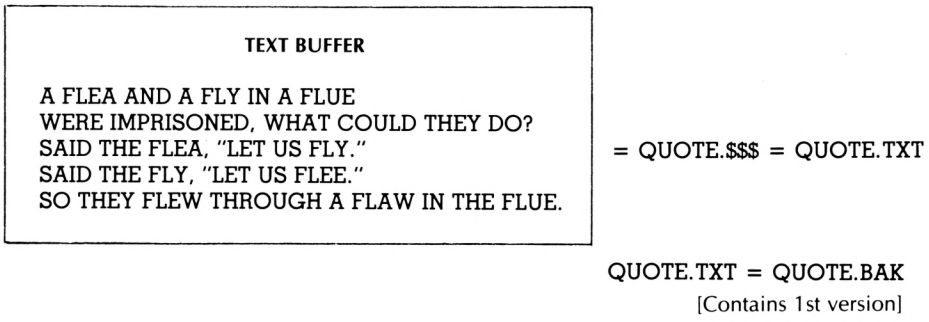
Find F the first comma (F, ^Z) and place the CP behind it; then display T the line from the CP.

```
2: *3D0TT<cr>
2: WERE IMPRISONED, WHAT COULD THEY DO?
2: *
```

Delete D the first 3 characters 3D and type the entire line 0TT. When the line is typed, the 3 characters S, O, and space are gone.

```
2: *B#T<cr>
1: A FLEA AND A FLY IN A FLUE
2: WERE IMPRISONED, WHAT COULD THEY DO?
3: SAID THE FLEA, "LET US FLY."
4: SAID THE FLY, "LET US FLEE."
5: SO THEY FLEW THROUGH A FLAW IN THE FLUE.
1: *E<CR>
A>
```

Move the CP to the head of the text buffer B and display all #T lines. The E command then ends the session, transferring text and renaming files as shown.



ED has its own set of commands which may be issued after its :* prompt. We have divided the functions of these commands into three categories for description under the headings referenced below:

- Moving text between diskette files and the text buffer (Text Transfer)
- Positioning the character pointer and displaying text (CP Control and Text Display)
- Deleting and adding text (Text Editing)

Some guidelines for combining ED commands are also provided. Use the Command Reference Card as an alphabetized summary of the ED commands. Where n appears before a command, it may be an integer value between 1 and 65,535. Special cases for the variable are 0 for n, which represents one-half of the text buffer capacity; and #, which represents all the lines from a file, or full buffer capacity.

ED TEXT TRANSFER

COMMANDS	FUNCTION
nA<cr> #A<cr> 0A<cr>	Transfer (by Appending) the specified number of lines (n) from the source file to the text buffer. The source file is identified in the command line issued to invoke ED. If # is specified for n, the buffer will be appended with the entire file, or a maximum of 65,535 lines; 0A will append the file to the buffer until the buffer is half full.
E<cr>	End the edit session by moving the text from the buffer and any remaining text from the source file into the file with the .$$$ extension. The

source file is renamed with the .BAK extension, and the .$$$ file is renamed with the extension of original source file. ED then exits, returning control to CP/M, which displays its A> prompt.

H<cr>

Transfer text from the buffer to the beginning of the .$$$ file; all text left in the source file prior to issuing the H command is also moved into the .$$$ file. The .$$$ file is renamed to become the latest version of the source file. The original source file is given a .BAK extension and a new, empty .$$$ file is created by ED. The H command saves all edited text and allows the user to continue editing the new source file (as if an E command had been issued, followed by reinvoking ED, so as to edit the latest version of the source file). The text of the source file is placed in the text buffer, and the user may continue editing operations.

Use the H command frequently if you are inserting large portions of text into the buffer. It ensures protection against losing text from the buffer due to operator errors or equipment failure. (If the power fails while in ED, all text in the buffer is lost.)

O<cr>

Ends the editing session by returning to the original source file, emptying the text buffer, and erasing the temporary (.$$$) file. The previous editing operations are ignored. When the O command is issued, ED responds by displaying (Y/N)? because all changes made to the text will be disregarded. A Y response will return you to the original source file; N will allow you to continue editing. Issue a #A command after a Y response to place the text of the source file back in the text buffer.

R<cr>

Reads the entire file created with the X command and transfers its text into the buffer at the current position of the CP. There is no limitation on the number of reads you may perform with the R command except to the point of filling the

text buffer. The transfer does not affect the file created with the X command, and ED assigns contiguous line numbers to the transferred text inserted before the CP. When you end an edit session with the E, Q, or ^C commands, ED erases the X command file.

Rfilename<cr>

This command functions like the R command, but the file read for the transfer operation may be any file with a .LIB extension. This command makes it convenient to create a "stock" document that, with a little editing, may be used repeatedly. For example, you can create a file that contains the standard form of an invoice, naming it INVOICE.LIB. Then, when you're ready to create a particular invoice, you could use this command to transfer the stock invoice into the text buffer for the altering that would make it individualized. Use of this command does not in any way change the contents of the .LIB file. Its image is merely placed in the text buffer.

Q<cr>

Quit editing and return to CP/M. Issuing this command will leave the original source file as it was before invoking ED and cause the temporary file to be erased. As with the O command, ED asks, "(Y/N)?"; type Y and ED proceeds to quit the session.

The O command should be used if you have incorrectly altered the source file and want to start over; use Q when you want to cancel previous editing operations and end the session.

---

nW<cr>
#W<cr>
0W<cr>

Write n lines of text from the buffer, transferring them to the temporary (.$$$) file. Transfer starts with the first line in the text buffer.

Transferred lines are placed behind any lines already contained in the .$$$ file. As the lines are transferred, they are deleted from the text buffer. W will not close the .$$$ file; it is primarily used to make more room in the text buffer.

#W will transfer the entire contents of the text

buffer to the .$$$ file, while 0W will transfer text until the buffer is left half full.

nX<cr> #X<cr>	Transfer a specified number (n) of lines from the text buffer (following the current position of the CP) into a special temporary file. The file created with the X command, often referred to as a *block move* file, is assigned an extension of .LIB. It would appear under a directory as X$$$$$$$.LIB but is active only during the current editing session and is erased when the E, Q, or ^C commands are issued. When you have transferred lines of text into the X command file, they may be read back into the buffer by issuing the R command. In this way, you may move large blocks of text to any place within the buffer (and hence to the new version of your source file) by repositioning your current line reference as indicated by the location of the CP. Lines of text transferred to the X command file are not removed from the buffer. When placed in the file, they immediately follow any previously transferred lines. When # is specified for n, all the lines following the CP to the end of the text buffer are transferred.
0X<cr>	When 0 is specified for n, the X command file is emptied and any efforts to read from the file using the R command will result in ED displaying the following error message: BREAK "O" AT where AT is the current position of the CP as indicated by ED's : * prompt.

ED CP CONTROL and TEXT DISPLAY

COMMANDS	FUNCTION
B<cr>	Position the CP at the beginning of the text buffer.
-B<cr>	Position the CP at the end of the text buffer.

nC<cr>	Place the CP forward a specified number (n) of characters.
-nC<cr>	Move the CP back a specified number (n) of characters. The Carriage Return and Line Feed which follow each line are counted as separate characters.
Fstring^Z<cr> nFstring^Z<cr>	Find the first occurrence of the characters following the F (represented here by string) and position the CP immediately after the last character. This command only searches for the string from within the text buffer and only from the current location of the CP. (n) represents the numbered occurrence of the string specified. 5Fthe^Z<cr>, for example, would search the text buffer for the fifth occurrence of "the" and place the CP between the "e" and the character immediately following it. When a Carriage Return is part of the string you wish to search for, substitute it with ^L. It's a good idea to issue a T command after your F command so you can be sure of the current position of the CP. Be sure the string you're searching for occurs after the CP.
nL<cr>	Position the CP a specified number (n) of lines forward.
-nL<cr>	Position the CP n lines back (toward the beginning of the text buffer).
0L<cr>	Put the CP at the beginning of the current line. Typing -#L<cr> will position the CP at the beginning of the first line from anywhere within the text buffer. #L<cr> will place the CP at the beginning of the line that follows the last line of text.
:m<cr>	Not actually a command in itself, but used to define a range of action for the effects of other commands, :m is employed as a prefix to show where a command should halt (where m is the line number). For example, you could use this prefix to have ED type out lines 5 through 15 from anywhere in the text buffer by issuing the

command 5::15T<cr>. The range defined by :m always starts at the CP when one line is specified in the command. For instance, :25T<cr> issued from the beginning of line 5 will cause lines 5 through 25 to be displayed, with the CP returning to the beginning of line 5.

---

n:<cr>

Position the CP at the beginning of the specified (n) line.

Nstring^X<cr>

Search the text buffer and the source file, if necessary, for the text specified in place of string, positioning the CP directly after it. The N command will automatically append lines from the source file if the specified character string cannot be found in the buffer. If the buffer space is filled by the lines appended from the source file in search of the text string, lines are moved from the buffer into the temporary file.

nNstring^X<cr>

Entering a number before N causes the command to find the numbered occurrence of the character string. The N command will search the entire source file for the specified character string, while the F command will limit its search to that part of the file in the text buffer. Remember that the search commences from the current position of the CP, which will be placed directly after the character string if it is found. See the F command description above.

P<cr>
nP<cr>
-nP<cr>

Place the CP one page ahead of its current position and display the number of pages (n) specified. The P command is a good way of moving the CP through the text buffer while having its contents displayed. A page of lines within the text buffer is 24 lines. This command first moves the CP and then displays the page following it. When - precedes the number specified for n, the CP is moved back.

0P<cr>

When 0 is specified, the page following the CP is displayed, but the CP retains its current position. This form of the P command is a convenient way of seeing what's in the text buffer without moving the CP.

T<cr> nT<cr> #T<cr>	Type (display) the line in which the CP is presently placed. Displayed text begins from the current position of the CP. When a number is specified for n, that number of lines will be displayed, beginning at the current line and position of the CP. When the # symbol is used for n, the entire text buffer contents below the CP will be displayed in the same manner.
0T<cr>	When you want to see the text of the line before the CP, use 0 for n.
0TT<cr>	This form of the T command will display the entire line in which the CP is presently placed without regard to the CP's position.
-nT<cr>	When a - symbol precedes the number specified for n, ED will move back through the buffer that number of lines and display all lines from there to the current position of the CP. To view the entire contents of the text buffer, use the B#T command sequence (see the B command).

-V<cr> V<cr>	The V command is used to control ED's line numbering display mode. The line numbers, helpful in locating text and CP positions, are not actually part of the text. They are displayed for convenience when you enter or append text into the text buffer. When you wish to see how the text of your file will appear without line numbers, or want to enter text in lines longer than allowable when the numbers, colon, and ED prompt are on screen, issuing the - form of the V command will turn off the line numbers. Line numbers may be restored by entering the single letter V command form.
0V<cr>	This form of the V command is used to display the number of bytes currently occupying the text buffer. This information is in the form of two numbers: the first number is the amount of unused space remaining (free); the second number represents the maximum number of bytes the text buffer can hold. By subtracting the first number from the second, you will know how many characters of the file you are working on are in the buffer. The numbers are displayed in this format:

29607/30427

This example shows that there are 820 characters of text currently in the buffer.

ED TEXT EDITING

COMMANDS	FUNCTION
nD<cr> -nD<cr>	Delete the specified number (n) of characters ahead of the CP. When - precedes the specified number, characters are deleted from behind the CP. When a number is not specified, the command deletes only the character ahead of (D) or behind (-D) the CP. A carriage return and line feed count as two separate characters.
I<cr>	Insert text into the buffer. This is the first ED command issued to create a new file. The characters you type following this command are accepted by ED and placed in the text buffer following the CP. To stop inserting text, hold down the CTRL key while typing Z (^Z); you'll know you're out of insert mode when ED's prompt returns to the display. While inserting text in the buffer via the I command, the following line editing commands may be used: ^H to delete last character typed; ^M for a carriage return; ^R to view the entire line currently being entered; ^U and ^X to delete the current line. Everything typed after the last carriage return is considered the current line while in insert mode. Remember that while you're in insert mode, ED will recognize everything you type as text to be added behind the CP; if you want to issue another ED command, type ^Z first.
Istring ^Z<cr>	When you need to insert a string of text in a line, use this command. It will insert the sequence of characters represented by "string" to the right of the CP and then stop insert mode (^ Z). Notice the space between the last character of the string and the ^Z.

Istring<cr>

This form of the I command is used to insert separate lines. The line of text (represented by string) is placed in the buffer following the CP. The carriage return <cr> that follows the string shows that it will become a separate line. This is the difference between this form of the command and the one shown above. There is no need to type a ^Z after this command. Remember that what you will be inserting will mark the end of a single line.

---

nJfstring^Zistring^Zestring^Z<cr>

The *juxtaposition* command is used to find fstring, insert istring after it, and then delete all text between istring and estring. Notice that a ^Z is used to separate each string of characters. When the command's operations are completed, the CP will be positioned directly after istring. The J command will be repeated for n occurrences of fstring.

As an example, if you wanted to change LET US FLY. to LET'S GET OUT OF HERE! in the QUOTE.TXT file shown earlier, the command line:

```
1: *JLET^Z'S GET OUT OF HERE! ^Z'' ^Z<cr>
```

could be used. ED would respond by finding the fstring LET in line 3 (its first occurrence), inserting 'S GET OUT OF HERE! after the T, and then deleting US FLY. You would see this take place by ED displaying 3: *, which tells you that the CP is now directly after ! in line 3. You could then examine the altered line using the T command as shown:

```
3: *0TT<cr>
3: SAID THE FLEA, ''LET'S GET OUT OF
   HERE!''
3: *
```

Proceed with caution when issuing multiples of the J command at one time (3J).

K<cr>
nK<cr>
-nK<cr>

Delete (Kill) the specified number (n) of lines from the text buffer. Deletion takes place after the CP unless - precedes the number specified.

-K by itself will kill all characters in a line before the CP.

0K<cr> Use this form of the command to kill all the characters in a line that are before the position of the CP.

---

nMcom1com2com3string^Z<cr>

The M command is used to repeatedly (n) execute a specified sequence of commands. Such a sequence of commands is called a *Macro*. The Macro is specified in place of comstring.

The Macro command MFSAID^Z-3DICREAMED^Z0TT<cr> uses the F, D, I, and T commands in sequence. In editing the QUOTE.TXT file, this command would repeatedly find SAID (putting the CP after the D), delete 3 characters behind the CP (AID), insert CREAMED, and then display each line being changed (0TT).

Execution of the command would appear as shown below (text includes changes made in the example of the J command):

```
1: *MFSAID^Z-3DICREAMED^Z0TT<cr>
3: SCREAMED THE FLEA, "LET'S GET OUT OF
   HERE!"
4: SCREAMED THE FLY, "LET US FLEE."
BREAK "#" AT ^Z
4: *
```

The BREAK#` AT ^Z error message is ED's way of telling you that it cannot execute your command the number of times specified. This is normal for the J command if 0 or 1 is specified for n, or if n is not specified at all. 4: * is displayed for two reasons in this case: first, because that's where ED encountered the error condition; and second, because the CP is now in line 4 (directly after the D).

---

nSfstring^Zreplace^Z<cr>

Substitute by finding and replacing; fstring is used here for the combination of characters to be found and replace for the characters that are to be used as the substitute. The command will operate on n occurrences of fstring.

ED searches the entire text buffer for fstring beginning from the CP.

The S command has the effect of consecutively issuing the Find, Delete, and Insert commands.

As you've seen in our examples and descriptions, ED's commands may be grouped into one command line. Combining commands saves time and makes editing less tedious, but there are some guidelines to follow.

1. Use the Carriage Return or Enter key only to end a command line.
2. Use ^Z to separate and end character strings within the F, I, J, N, and S commands.
3. Use ^L to represent a <cr> in a character string.
4. You may not use more than 100 characters in command lines for the J, F, N, and S commands.
5. So that you won't suffer the consequences of potentially catastrophic typing errors, the E, H, O, and Q commands must be entered alone.

There are two rules you should always follow before you begin an edit session: First, before you start, be sure the diskette you are going to use is not write-protected. Second, use STAT to determine that your diskette will be able to hold your text, whether you're just revising a file or creating a new one.

For example, if you want to revise a file that currently occupies 1 kilobyte (STAT will show this as 1K—the equivalent of 1,024 bytes), you should check that your diskette will be able to contain at least 2K more. This is because the .BAK file will take up 1K by itself and your revisions may take an additional 1K.

When creating a new text file, figure about 3 kilobytes for every page of type (single spaced, 66 lines per page).

ED ERROR MESSAGES

The following messages are displayed when the described error conditions are encountered by ED.

FILE IS READ/ONLY

You'll see this message if you're trying to edit a file on a write-protected

diskette, or a file that's been set to R/O with STAT. When you attempt to end the edit session, the message below is displayed.

BDOS Err ON x: file R/O

This error message is displayed when you attempt to edit a file on a write-protected diskette or when the file has been set to R/O with STAT. It comes after the error message shown above. The system will stop after this message until you press the <cr> key to return to CP/M's A> prompt. If the diskette was write-protected, ED will not be invoked; if the file was set to R/O, the changes made in the text buffer are lost and will not be written to disk. For the latter case, you can reset the file, using STAT, to R/W.

DISK OR DIRECTORY FULL

This error message can mean two things—you have started ED without specifying a file name, or there is not enough space on the diskette to hold the text being transferred from the buffer. Always use STAT before beginning an editing session to make sure you'll have room on the diskette. If this message is displayed because you're out of room, you'll have to use the REN command to rename the .BAK version of your file and start all over again. If you're creating a new file in this situation, you'll have to use another diskette and start at the beginning.

BREAK "y" AT
x: *

where y is one of the following error characters and x is the line number where ED encountered the error condition.

? ED does not recognize what you have typed as a valid command.

> The text buffer is full. The user must issue one of the following commands:

D, K, N, S, or W

to transfer text from the buffer. The > character is also displayed by ED when the text contained in an F, N, or S command is too long (> 100 symbols).

ED cannot execute your command the number of times specified. For example: 10FEX <cr>, when EX occurs only 3 times in the text buffer.

PIP

PIP is an acronym for *P*eripheral *I*nterchange *P*rogram. You will be using this utility often; it allows you to move and copy files from one system device to another.

With PIP, you can transfer a copy of a file from one disk drive to another, rename it during the move, consolidate files in a specified order, and begin and end copying from specified locations within files.

The system list device can also be sent files for output through PIP. When the console is specified as a system device, you may create files from your typed keyboard input.

There are two ways of invoking PIP:

1. The first is by typing PIP and defining the particular transfer operation you want performed, ended with a "<cr>." The general format of defining the transfer operation is *destination=source*:

   A> PIP B:=A:filename.ext<cr>

   where B: specifies that disk drive B is to be the destination for a copy of the file filename.ext residing on the source drive A:. This operation will not affect the source file, it will remain on drive A. Remember not to type spaces around the equals sign—PIP doesn't like it.

   When you wish to copy all the files of a certain type from one diskette to another, use an ambiguous file reference:

   A> PIP A:=B:*.ext<cr>

   where .ext is the file type. PIP will display the name of each file of the type .ext as a copy of the file is being transferred from drive B to drive A.

2. The second method is often preferred when complicated or multiple transfer operations are to be specified. Simply type PIP<cr>. This causes PIP to be loaded into the Transient Program Area. PIP will then signal you that it is ready by displaying its * prompt, at which time you may type in the definition of the transfer operation you wish performed. After each operation is performed, you may specify another. When you want to return to CP/M after using PIP in this manner, enter ^C.

   Display 2-10 shows PIP being invoked in this manner.

Notice that the *source* side of the transfer operation is not specified. This causes PIP to assume that the default drive contains the source files. The default drive for PIP is the drive that contains the diskette from

```
A>PIP<cr>
*B:*.TXT<cr>

COPYING-
MEMO.TXT
QUOTE.TXT
REPORT.TXT
TEST.TXT
*B:=ED.COM<cr>
*^C

A>
```

Display 2-10

which PIP was invoked. For example, if the diskette in drive B contains a copy of PIP, and drive B is selected before invoking PIP (B:<cr>), PIP will assume drive B contains the source files.

To rename a file during its transfer, merely use the new name on the *destination* side of the PIP command line. For example, suppose you want to transfer a copy of the file NOTE.TXT from drive A to drive B and rename it TEXT.NTE:

```
A> PIP B:TEXT.NTE=A:NOTE.TXT<cr>
```

When you want to consolidate copies of two or more files into one file, use a command line like the one shown below:

```
A> PIP B:COSTS.MAY=A:INVOICE.ONE,
    B:INVOICE.TWO<cr>
```

This command line would result in PIP creating a new file on drive B called COSTS.MAY from INVOICE.ONE on drive A and INVOICE.TWO on drive B. COSTS.MAY would be a consolidation of INVOICE.ONE and INVOICE.TWO, in that order.

To begin and end copying from specified locations within a file requires the use of the S and Q PIP parameters:

```
A> PIP B:=A:NOTE.TXT[SMONEY
    ^ZQPAPER^Z]<cr>
```

This command line requests PIP to transfer a copy of the file NOTE.TXT to drive B from drive A, *S*tarting with the characters MONEY and *Q*uitting directly after PAPER. The starting and quitting character strings are defined behind S and Q, respectively, and each is ended with ^Z. The S and Q parameters are more fully described under PIP Parameters below.

The system list device is identified by CP/M as LST:; this is usually the printer attached to your computer (see the description of STAT). When you use LST: as the destination device in a PIP command line, the specified file will be printed. Of course the printer must be ready to operate (have paper and power on, et cetera). Such a command line would appear as:

A> PIP LST:=B:NOTE.TXT<cr>

The same process can be applied to the computer's console display:

A> PIP CON:=B:NOTE.TXT<cr>

CON: identifies the display, so a copy of NOTE.TXT will be displayed on the screen. This PIP command line provides the same function as the built-in command TYPE.

When CON: is used in the source side of a PIP command line, you can create text files from typed keyboard input. This convenient way of creating small text files that you probably won't need to revise, makes use of CP/M's line editing commands. Below, we have depicted a user creating the file NOTE.TXT in this manner.

```
A> PIP NOTE.TXT=CON:<cr>
 ^J
 ^I THINGS TO REMEMBER:^J^M
MONEY FOR TICKETS^J^M
UMBRELLA^J^M
KEYS TO CABIN^J^M
BOOKS AND PAPER^J^M
FEED THE CAT^J^M
 ^J
 ^Z

A> TYPE NOTE.TXT
     THINGS TO REMEMBER:
MONEY FOR TICKETS
UMBRELLA
KEYS TO CABIN
BOOKS AND PAPER
FEED THE CAT

A>
```

When PIP processes the command line, CP/M's prompt does not reappear. This indicates that PIP will accept typed entries as input for the file NOTE.TXT. The ^J line editing command produces a line feed, used in the file as a blank line. ^I is used to indent the first line of the file. Notice the ^J ^M sequence after each line of type: ^J positions the cursor on the next line, and ^M performs a carriage return.

When all lines have been input, the user enters ^Z. PIP accepts this as the end of the file. The file is closed, and control of the console is returned to CP/M. The TYPE command is then issued so the file will be displayed on the console screen.

PIP PARAMETERS

You may qualify the operation of PIP by including parameters in a command line. PIP parameters are single characters separated in the command line by square brackets ([ ]) following the file name or device:

PIP B:=A:filename.ext[EV]<cr>

Notice that you can specify more than one parameter. The E parameter instructs PIP to display the contents of a file as it is being transferred; the V parameter is used to verify that the copy of a file matches its source.

Individual parameters and their functions are described below.

Parameter	Function
[Dn]	Delete characters extending beyond column n. The specified column is counted from the last <cr> in each line of the file. When column n is counted, all characters past it are ignored by PIP until another <cr> is read. Use this parameter to truncate the lines of a file that are too long for a narrow console display or printer.
[E]	This parameter may only be useful when files containing readable text are being transferred. It will echo to the console (display) transfer operations as they are being performed by PIP.

[F]	Ignore form feed indicators (used to define page breaks) in the source file. This parameter may be useful when your printer does not respond to form feed characters, or when you wish to view the file from your console screen. The P parameter can be used in the same PIP command line to reformat pages with a specified number of lines per page ([FP50]).
[Gn]	Copy the file(s) from user area n. See the description of the USER command in Section 2.3.
[L]	Change all upper-case characters A through Z to lower-case during the file copy transfer.
[N] [N2]	Number each line of the file being transferred, beginning with 1. Colons follow the numbers and leading zeros are suppressed (1: TEXT). Specifying 2 after N will include leading zeros and replace the colons with spaces (000001 TEXT).
[O]	This parameter is used when transferring an object file (i.e., binary data or program files). Use of the O parameter is not necessary for .COM file transfers or files that contain readable text. This function is provided to prevent PIP from interpreting data that equals ^Z as the end of the file. PIP reads ^Z as an end of file mark for transfer operations.
[Pn]	Specifies the number of lines n per page. A page break will occur after every n line. If n = 1, or is not specified, page breaks will occur after every 60 lines.
[Qstring^Z]	Quit copying from the source file or device when the specified sequence of characters (represented here as "string") has been encountered. The Quit string must be ended with a ^Z. If the Quit string specifies lower-case characters, PIP must be entered before the command line is issued. Using the S and Q parameters together, you

	are able to define portions of a source file you may need as a separate file.
[R]	Copy a file which has been set by STAT to SYS. Normally, PIP will not attempt to read this type of file.
[Sstring^Z]	Start copying from the source file only when the specified sequence of characters is encountered (represented here as string). As with the Q parameter, the string must end with a ^Z, and PIP must be entered before the command line is issued if lower-case letters are used in the string.
[Tn]	Set or expand tab positions to every nth column of the file during transfer of the copy.
[U]	Change all lower-case characters a through z to upper-case during file copy transfer.
[V]	Verify that the copy of the file transferred is the same as the original file. PIP does this by rereading portions of the transferred copy after they have been written. The P parameter operates only when the destination is to be a disk file.
[W]	Copy into or overwrite a Read/Only (R/O) file. Normally, CP/M would not allow PIP to write into a file that has been set to R/O status with STAT. When such an attempt is made, the message: DESTINATION IS R/O, DELETE (Y/N)? is displayed. The transfer would not occur unless a "Y" response is typed. The W parameter allows you to avoid this interaction.

ERROR MESSAGES

The following messages are displayed when an error condition is encountered by PIP.

SEEK ERROR - DRIVE x TRACK 2

If you attempt to PIP a file copy to an unformatted or otherwise unreadable diskette in drive x, you'll see this message displayed. It will be followed by:

Bdos Err On x: Bad Sector

Type ^C to recover from this error condition.

Bdos Err On x: R/O

This error message is displayed when you direct PIP to transfer a file to a diskette that has been set to Read/Only with STAT. Type ^C to recover.

CANNOT CLOSE DESTINATION FILE: = filename.ext

This error message is displayed when you attempt to write a file to a diskette that has a write-protect tab affixed to it.

DESTINATION IS R/O, DELETE (Y/N)?

Any file that is not protected can be destroyed by PIP. This can occur when you direct PIP to transfer a file with the same name as the destination, in which case PIP will overwrite the destination file.

If the destination file was previously protected by setting it to Read/Only with STAT, you'll see this message displayed. The exception is when you use PIP's W parameter, which is described on the previous page.

NO FILE: = filename.ext

PIP cannot locate the specified source file. Use the DIR command to be certain the file resides on the source diskette before issuing a PIP command.

QUIT NOT FOUND: =filename.ext[Qstring]

You'll see this message displayed when PIP is unable to locate the character string in the source file specified with the Q parameter (see the description of the Q parameter above).

START NOT FOUND: = filename.ext[Sstring]

The character string specified with the S parameter cannot be located in the source file.

STAT

This description of STAT is in two parts: Part I describes STAT as it pertains to files; Part II shows how STAT is used to control system device assignments.

I. File Information

STAT provides information on the STATus of CP/M files—their size, type, number, and the amount of space remaining on a specified disk. This last item is important (you should know how large a file is *and* how much room is unallocated on a disk you may want to transfer the file to).

STAT offers the user six options related to a request for file information. These options are entered through the keyboard in the STAT command line which is ended, like all transient command lines, by depressing the Carriage Return (<cr>) key. The file information options are shown below, followed by examples of their usage.

STAT<cr> When STAT is typed after the A> prompt (or B>, or C>, etc.), and is followed by a carriage return, STAT will display in bytes the amount of unused storage remaining (and available) on all the currently logged disk drives.

STAT x:<cr> This command line is similar to the first, except now the user wishes to know how much available space is on disk drive x (x can be any valid disk drive).

STAT *.xxx<cr> When this command line is entered, STAT will respond by displaying information on the group of files of the type that satisfy .xxx, used here to mean any unambiguous extension. This may be, for instance, all .DAT type files, in which case STAT will display information on the size of all .DAT files on the currently logged disk in bytes and records. This command line can be altered so that the user gets information about .xxx files on a specified disk (STAT B:*.xxx).

STAT ufn<cr> Here the user wants to know about a specific file (*u*nambiguous *f*ile *n*ame). This command line can also be altered to become a request for

information about a specific file on a specified (it must be valid) disk drive: STAT x:ufn. Remember to enter both the file name and type. STAT B:NOTE will only display the error message NO FILE; STAT B:NOTE.TXT will display useful information.

STAT *.*<cr> Entering this command line requests STAT to display a more informative type of file DIRectory. STAT will display information in Records, Bytes, and Extents for all the files contained on the currently logged-on disk drive. STAT will display the file name and type and the amount of unused space available. This command line can also be refined when the user specifies a valid disk drive (STAT x:*.*).

STAT x:=R/O<cr> Enter this command line when you wish to designate an entire drive as Read/Only. Once this command line has been entered, only warm-booting the system can reverse its effect (this is easily accomplished by issuing the ^C command).

The first command line in Display 2-11 shows the user requesting STAT to display the amount of available space currently on drive B.

```
A>STAT B:<cr>

  BYTES REMAINING ON B: 120K

A>STAT A:*.DAT<cr>

  RECS     BYTES   EXT   ACC
    25        4K     1   R/W  A:ABC.DAT
    16        2K     1   R/W  A:DEF.DAT
    27        4K     1   R/W  A:GHI.DAT
    65        9K     1   R/W  A:BET.DAT

  BYTES REMAINING ON A: 141K

A>
```

Display 2-11

STAT informs the user that there are 120K bytes of unused space on drive B.

In the second command line, the user wants to know the size of all the .DAT files on drive A. RECS is the number of records in each .DAT type file, BYTES the number of bytes (rounded up) of the particular file, and EXT the number of extents each .DAT file contains. An extent is a measurement CP/M uses to manage files; you can think of an extent as a block of data equivalent to 128 records. Records can be thought of as blocks of data in 128-byte units. ACC means access, R/W denotes that the file can be read and written to, and R/O that the file has been set to Read/Only.

STAT allows you to set the access of a specified file to four different attributes:

R/O Read/Only—the file can only be read. This is a good way to protect a file on a diskette that you don't want to write-protect with a tab. In this way you are able to write to other files on the diskette and still guard particular files from being corrupted.

R/W Read/Write—the file can be both read and written to. This is the normal attribute setting for CP/M files.

DIR Directory—the file will be displayed in response to a DIR command. This is the normal attribute for CP/M files.

SYS System—the file will not appear in a DIRectory of files. When this attribute is assigned to a file, it will be hidden from being displayed by the DIR command; it will, however, appear in parentheses in response to a STAT request for file information (STAT *.*<cr>).

Attempts to transfer a copy of a SYS file with PIP will result in the error message:

NO FILE: =file.name

unless PIP's R parameter is specified. A file set to the SYS attribute can be removed from secrecy by resetting it to DIR.

You can set a file attribute with the STAT command line shown below:

A> STAT filename.ext $att

where att is one of the attributes described above. Notice the $ symbol;

this is what instructs STAT to set the attribute. When STAT has processed the command, it will display:

filename.ext Set To att

Ambiguous file references may be used:

A> STAT *.* $att<cr>

in which case STAT will display the name of each file as being set to the specified attribute.

If you should forget the possible attribute settings for CP/M files, type the command line shown in Display 2-14. It causes all the valid options for attribute settings to be displayed.

STAT will display the current user status when the command line

A> STAT USR:<cr>

has been entered. STAT's response to this command is described under the "Finding Active User Areas" heading of the USER command in Section 2.3 (page 31).

II. Device Assignment

In order to provide flexibility in Input/Output device management, CP/M uses logical device names for the I/O devices. The software in the BIOS portion of CP/M contains logic that is used to control (drive) the peripheral devices. The software module (subroutine) that controls the printer may also be capable of driving a console. This is where STAT comes in—it enables you to assign four logical names to actual devices so that a particular driver subroutine will be controlling a particular physical device. These are:

CON:	the facility for controlling CONsole functions, such as entering commands and displaying information (both input and output)
RDR:	the ability to receive information—it comes from Paper, Tape, Reader (only input)
PUN:	the output function—derived from Paper, Tape, Punch (only output)
LST:	the printing (list) function (only output)

There are 12 actual devices that may be assigned to the logical names with STAT:

TTY: Teletypewriter/slow-speed console device	**CRT:** Cathode Ray Tube/high-speed console device
BAT: Batch processor	**UC1:** User defined Console
PTR: Paper Tape Reader	**UR1:** User defined Reader (No. 1)
PTP: Paper Tape Punch	**UR2:** User defined Reader (No. 2)
LPT: Line Printer	**UP1:** User defined Punch (No. 1)
UL1: User defined List device	**UP2:** User defined Punch (No. 2)

STAT displays the possible logical-to-physical device assignments under the IOBYTE ASSIGN: heading of the information shown in Display 2-14.

The current logical-to-physical device assignments are revealed when the command line shown in Display 2-12 is typed into the console.

```
A>STAT DEV:<cr>

COM: is CRT:
RDR : is TTY:
PUN : is TTY:
LST  : is LPT:

A>
```

Display 2-12

The device assignments will vary among systems, so don't be alarmed if what you see displayed in response to this command is different from what's shown in Display 2-12.

The command line:

STAT x:DSK:<cr>

directs STAT to display some statistics concerning a disk drive (where x is a valid drive designation). STAT's response to this request is shown in Display 2-13.

```
A>STAT A: DSK:<cr>

        A: Drive Characteristics
     2048: 128 Byte Record Capacity
      256: Kilobyte Drive Capacity
      128: 32 Byte Directory Entries
      128: Checked Directory Entries
      256: Records/Extent
       16: Records/Block
        2: Reserved Tracks

A>
```

Display 2-13

```
A>STAT VAL:<cr>

TEMP R/O DISK  : D:=R/O
SET INDICATOR: D:FILENAME.TYP $R/O $R/W $SYS $DIR
DISK STATUS    : DSK: D:DSK:
USER STATUS    : USR
IOBYTE ASSIGN:
CON: = TTY:  CRT:  BAT:  UC1:
RDR: = TTY:  PTR:  UR1:  UR2:
PUN: = TTY:  PTP:  UP1:  UP2:
LST : = TTY:  CRT:  UL1:  UL1:

A>
```

Display 2-14

Of course, the numbers shown in Display 2-13 are going to vary, depending on the system you are using.

The command STAT VAL:<cr>, as shown in Display 2-14, will cause STAT to display the valid options for STAT command lines, except those used to get statistics on files (described on page 57).

For example, TEMP R/O DISK: D:=R/O is an abbreviation for the command line used to set an entire drive to Read/Only status:

STAT B:=R/O

STAT ERROR MESSAGES

The following messages are displayed when an error condition is encountered by STAT.

SEEK ERROR - DRIVE x TRACK 2

If you are using an unformatted or otherwise unreadable diskette, you'll see this message displayed. It will be followed immediately by:

Bdos Err On x: Bad Sector

You'll have to warmboot the system to recover: type ^C.

FILE NOT FOUND

This is usually the result of a typing error. You may not have included a valid drive designation, or you forgot to use a colon with it, or you misspelled the file name. This message can also mean that the file you requested is not on the drive you requested.

INVALID FILE INDICATOR

If you attempt to set a file attribute to anything except R/O, R/W, SYS, or DIR this message will be displayed. If you have specified a valid attribute setting and this message is displayed, check to see if you included the $ symbol. This message may also appear if you put a space where it doesn't belong in a STAT command line.

Bdos Err On x: Select

You have specified a nonexistent disk drive. Typing ^C will return you to CP/M.

INVALID DISK ASSIGNMENT

This message is displayed in response to an error in the STAT x:=R/O command line. It may be the result of a typing error or an attempt to assign a disk drive to anything but Read/Only.

SUBMIT

SUBMIT allows you to submit for execution a series of commands that have been stored in a previously created file. Such a file may be created with ED and is always assigned an extension of .SUB.

When SUBMIT is invoked, it searches for the .SUB file named in the command line and creates a temporary ($$$) file. SUBMIT will move the commands to be executed from the .SUB file into the $$$.SUB file and execute them one at a time.

Below is an example of using the SUBMIT utility:

1. Create a file named FIRST.SUB with ED:

   A> ED FIRST.SUB<cr>

   When ED displays NEW FILE enter the following lines:

   I<cr>
   1: DIR<cr>
   2: STAT *.COM<cr>
   3: STAT VAL:<cr>
   4: ^Z
   :*E<cr>

2. When the FIRST.SUB file has been created, type the SUBMIT invocation command line:

   A> SUBMIT FIRST<cr>

In the directory displayed in response to the commands contained in FIRST.SUB, you'll also see $$$.SUB.

SYSGEN

SYSGEN stands for SYStem GENeration, the process of transferring a copy of the CP/M operating system to another diskette. Included in this transfer are the CCP (which contains CP/M's built-in and line-editing commands), the BIOS and BDOS. The transient commands, though they are utilities of the CP/M operating system, are not transferred during a SYSGEN operation. The .COM files containing the individual transient commands can be copied to another diskette using the PIP command, but the operating system is not recorded on the diskette in the form of a file. SYSGEN will allow you to transfer the operating system as you would a file.

Your primary reason for using SYSGEN will be to create bootable diskettes. Without an operating system, a diskette cannot be coldbooted or warmbooted. Use SYSGEN to place an operating system on any diskette from which you intend to perform a coldboot or a warmboot.

The MOVCPM and DDT utilities can be used in conjunction with SYSGEN to modify the operating system before transferring it. For complete instructions on these operations you can consult Digital Research manuals, "Introduction to CP/M Features and Facilities," "CP/M 2.2 User's Guide," and "CP/M Dynamic Debugging Tool (DDT)." We will discuss only how to transfer an unmodified version of the operating system.

Display 2-15 shows the series of steps involved in a SYSGEN operation.

```
A>SYSGEN<cr>

SYSGEN VER 2.2

SOURCE DRIVE NAME (OR RETURN TO SKIP) A
SOURCE ON A, THEN TYPE RETURN <cr>

FUNCTION COMPLETE

DESTINATION DRIVE NAME (OR RETURN TO REBOOT) B
DESTINATION ON B, THEN TYPE RETURN <cr>

FUNCTION COMPLETE

DESTINATION DRIVE NAME (OR RETURN TO REBOOT) <cr>
```

Display 2-15

Place your work copy of CP/M in disk drive A. When you type SYSGEN<cr> CP/M loads the SYSGEN program into the TPA and begins executing it. As operation starts, the version number of your particular SYSGEN program will appear on the screen (Line 1 in Display 2-15).

SYSGEN first directs you to type the "source drive name," (Line 2). This is the drive which contains the diskette with the operating system you want to transfer. Your system diskette is in drive A, so type "A". Notice that no carriage return is necessary; the next direction is immediately displayed.

Line 3 directs you to place the source diskette in drive A (which you have already done), then type a carriage return. When the return has been entered, SYSGEN will prepare a copy of the operating system for transfer to another diskette. FUNCTION COMPLETE will be displayed when the copy is ready (Line 4).

Next, Line 5 asks you to type the "destination drive name." This is the drive containing the diskette which is to receive the copy of the operating system from drive A. Our example shows that drive B was chosen.

As shown in Line 6, SYSGEN then directs you to put the destination diskette in drive B and type a carriage return. This destination diskette must be correctly formatted. Any files on the diskette will not be affected by the operation. However, if there is an operating system already on the diskette, it will be replaced by the one you are transferring.

When a copy of the operating system has been transferred, the FUNCTION COMPLETE message is displayed, and you are given the option of transferring the system to still another diskette (Lines 7 and 8, respectively). You will have this option after every additional SYSGEN operation.

Type a carriage return to end the SYSGEN operation and exit the program (OR RETURN TO REBOOT). You will be returned to the CP/M system in drive A.

You should always test any newly transferred operating system by performing a coldboot and a warmboot on the destination diskette.

ASM

The ASM program is CP/M's assembler. It is used to convert an 8080 assembly language program (ASM file) which you may have created with the ED program or another editor into a hexadecimal (.HEX) file. This HEX file is processed with the LOAD command to form a .COM file which your computer will be able to execute under CP/M.

The use of ASM is not described in this introductory guide. Should you require complete information on ASM, consult the Digital Research manuals, "CP/M: An Introduction to Features and Facilities" and "CP/M Assembler User's Guide," or the wealth of information contained in Thom Hogan's "Osborne CP/M User Guide."

DDT

The Dynamic Debugging Tool is used to *debug* a file (find any errors in a program and then correct them).

Because this book is intended as an introduction to CP/M for the beginning user, our description of DDT is limited to its use in placing a

file in the Transient Program Area of CP/M so that it may then be assigned to a separate, logical user area (see the USER command in Section 2.3).

For complete information on the DDT command, consult the Digital Research manual, "CP/M Dynamic Debugging Tool (DDT) User's Guide."

LOAD

LOAD will read a hexadecimal (.HEX) file using ASM and turn it into object code which will appear as an executable (.COM) file. The .HEX format machine code file is not altered and remains on disk as it was before LOAD was invoked.

LOAD is initiated by typing LOAD, followed by the name of the .HEX file which is to be the source of the newly created .COM file.

LOAD is primarily intended for use by programmers creating new utilities that will run under CP/M. Detailed information on the use of the LOAD utility may be found in "An Introduction To CP/M Features and Facilities."

MOVCPM

The MOVCPM utility is used to reconfigure versions of CP/M for use in computer systems with particular memory allotments. MOVCPM reconfigures the CP/M system to manage a specified amount of memory within the computer by creating an image of the operating system which may be saved on disk or allowed to take over control of the computer system and not be saved.

Storing the result of MOVCPM is handled with the SYSGEN and/or SAVE commands.

MOVCPM is generally used as part of the CP/M installation process, and because the intended audience of this book should have access to an installed system, we do not describe the operation of the utility here.

Should you care to know more about MOVCPM, consult "An Introduction To CP/M Features and Facilities."

XSUB

XSUB gives increased power to the SUBMIT utility by allowing the commands in the SUBMIT file to accept (even expect) user input from the console.

We do not feel that the XSUB command will be of any real use to the beginning user. As you become more proficient in the use of CP/M and wish to know more about XSUB, you'll find detailed information on it in the Digital Research manual, "CP/M 2.2 User's Guide."

2.5 GENERAL ERROR MESSAGES AND CONDITIONS

An *error condition* occurs when CP/M does not recognize the command you have issued or cannot perform a function due to some prohibiting factor. Errors typically result when a command is typed incorrectly or when CP/M cannot locate a file (such as a transient program) needed to complete an action. Usually, CP/M will display an *error message* when such a condition occurs. Error messages tell you what has happened to inhibit CP/M operation, so that you are able to correct the condition.

This section describes some of the more commonly encountered error messages—what they mean and how to recover from the conditions that cause them.

xxx?

This error message occurs when CP/M does not recognize what you have typed as a valid command. The xxx in the above example stands for the command as typed. After the message has been displayed, the A> prompt will reappear on the screen.

There are a couple of reasons why this message would appear. First, you may have typed the command incorrectly. For example, if you typed SRAT instead of STAT (a valid CP/M command) the error message would look like this: SRAT?. In this case, you would simply retype the command. The second condition that causes this message to be displayed is when the command is the name of a valid .COM file, but the file is not on the disk you specified. Again, if you attempted to issue the STAT command, but the diskette from which you issued it did not contain the file STAT.COM, CP/M would not recognize what you typed as a valid command. The only way you could get CP/M to recognize the command would be to designate a drive where the STAT program was present.

File Not Found or NO FILE

Both of these messages mean that the file (not the command) designated in the command line cannot be found on the designated diskette. The file you named could be ambiguous or unambiguous. The message may also appear if you have not typed a colon where required in the command line. The A> prompt reappears on the screen after the error message is displayed. To recover from this error condition check the following items:

1. Did you type the file name and extension correctly?
2. Have you neglected to type a colon where required?
3. Did you designate the correct disk drive?
4. Does the diskette in the drive you designated contain the requested file?

When you know what went wrong, retype the command, being sure not to duplicate the error.

Bdos Err On x: R/O

This message indicates that you have attempted to transfer data to or erase data from a diskette that has been protected against these actions. The x stands for the disk drive which contains the protected diskette. A diskette may be physically write-protected as described in Section 1.3 "Diskette Care" (page 5), or "logically" write-protected using the STAT command (see Section 2.3, page 20).

The A> prompt will not automatically reappear after this error condition has occurred. To recover, simply perform a warm boot by typing ^C.

Bdos Err On x: Select

If in a command you have designated a disk drive that CP/M cannot access, this message will be displayed. The x stands for the invalid drive. For example, if you issued the command DIR E: and your computer system has only drives A through D, then the following message would appear:

Bdos Err On E: Select

To recover from this error condition, type ^C.

Bdos Err On x: Bad Sector

This message indicates that CP/M is unable to process your command due to:

1. A failure in the electronics controlling the disk drive unit.
2. A faulty or excessively worn portion of your diskette's recording media.
3. When CP/M attempts to perform disk input/output operations on a diskette which was previously used in a disk drive unit not compatible with your own computer system.

You should first try to find out which of the above cases has caused this message to be displayed. Check the diskette and be sure it's compatible with your system. Try the same command operation after replacing the suspected diskette with one you know to be in good working order and typing ^C. If the message reappears, it was most likely caused by the first case. If it does not reappear, the first diskette was probably in poor shape.

Although the first diskette may be in poor condition, it may still be usable for recording data on another area of its media. Try reissuing the command using the same diskette. Should the error message recur on repeated attempts, give up—you've got a bad diskette that should not be used. Make a backup copy of anything you want saved from the diskette and then discard the bad one.

DRIVE X NOT READY - TYPE ANY KEY TO RETRY - CONTROL-C TO REBOOT

This message (or one of similar wording) may appear when you attempt to address a disk drive that does not contain a diskette, one whose door has not been closed, or one to which power has not been applied.

To recover from this error, make sure the drive is powered on, that there is a diskette in the drive, and that the door is closed. Then press any key to retry the command operation or type ^C to restart the system from drive A.

3 WORK COPY PREPARATION

To ensure that you always have an accurate copy of the software you purchased, a work copy should be made and used in place of the original end user diskette. The original should be write-protected immediately and stored in a place which meets the requirements specified in Section 1.3 "Diskette Care" after you have made your work copies.

Most distributors of CP/M include some kind of formatting utility, and many add a high-speed copy utility to the system diskettes. These will appear as .COM type files; the diskette formatting utility will typically appear as FORMAT.COM, and the fast copy utility will have a name similar to FSTCOPY.COM. If you find that the system diskette you have purchased does not contain a diskette formatting utility, ask your vendor where you can obtain diskettes with the proper CP/M format for your particular system.

Producing work copies involves three steps: activating CP/M, formatting a new diskette, and duplicating the original CP/M end user diskette. Remember, any new diskette you want to use with CP/M must have the correct format.

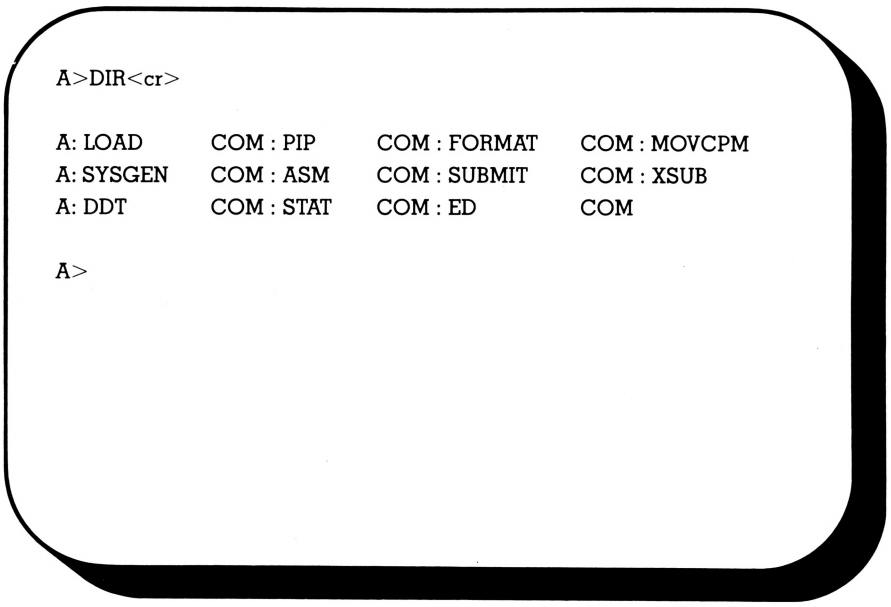
3.1 BOOTING CP/M

Follow the procedures described in Section 1.4; when the A> prompt appears on the display, you may begin executing commands through the operating system.

The first thing you'll want to do is find out what's on your diskette by typing the command:

DIR<cr>

Display 3-1 shows the information displayed in response to a request for a directory of files (there may be a slight variation depending on the exact release of your CP/M system and where you purchased it).



Display 3-1

3.2 DISKETTE FORMATTING

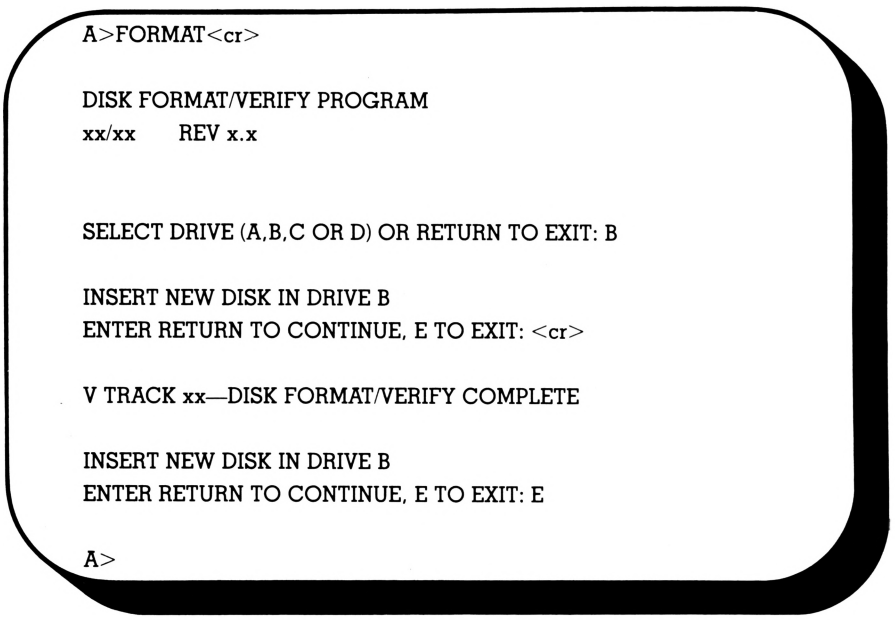
Before data can be transferred from the software master (in this case your original CP/M system diskette) to a new diskette, the new diskette must be properly formatted. *Formatting a diskette will destroy all data on it.* Be sure the diskette to be formatted is new or contains data you do not wish to keep.

To begin formatting a diskette using a formatting utility, simply type the utility's name (FORMAT is used here) and a carriage return after CP/M's A> prompt. Display 3-2 shows user interaction with a typical formatting utility.

Because the user's system diskette (containing the format utility) is in drive A, drive B was selected. The INSERT NEW DISK message was displayed in response. Next, a new diskette was inserted in drive B and the user entered a carriage return. When the utility formatted and then verified the last track of the new diskette, it presented the user with the option of formatting another diskette or returning to CP/M.

3.3 DUPLICATION

Now that you have formatted a new diskette, you're ready to create a copy of your original CP/M system. To accomplish this, you may use the fast copy utility if your system diskette contains one, or CP/M's SYSGEN and PIP utilities.



Display 3-2

SYSGEN will place a copy of the original system (which was put in drive A for the formatting operation) on a diskette selected by the user. This copy will contain the operating system and its built-in commands, but not the transient commands or any other files of any type. Referred to as a system disk or "system image," the product copy of SYSGEN will load like your original diskette.

Use PIP to transfer copies of the transient commands (or any other file types) onto the system image diskette created by SYSGEN.

Display 3-3 depicts a user's interaction with SYSGEN and PIP in the creation of a work copy.

The user has invoked the SYSGEN utility by typing SYSGEN<cr>. When SYSGEN is loaded from the diskette in drive A, it asks for a source drive to be designated. This may be any drive that has a diskette with the CP/M system in it. Here the user has entered A. SYSGEN then requests the name of the drive to which the system will be written. Because drive B contains the newly formatted diskette, the user has entered B and SYSGEN displays a reminder message so the user is sure of which diskette will have the system image placed on it. When the system image has been written on the diskette in drive B, SYSGEN displays the DESTINATION DRIVE NAME message again; this enables the user to create more copies of the system without having to reinvoke SYSGEN. By typing a carriage return in response to this request, the user is returned to CP/M in drive A.

```
A>SYSGEN<cr>

SYSGEN VER 2.2

SOURCE DRIVE NAME (OR RETURN TO SKIP) A
SOURCE ON A, THEN TYPE RETURN <cr>

FUNCTION COMPLETE

DESTINATION DRIVE NAME (OR RETURN TO REBOOT) B
DESTINATION ON B, THEN TYPE RETURN <cr>

FUNCTION COMPLETE

DESTINATION DRIVE NAME (OR RETURN TO REBOOT) <cr>

A>PIP B:=A:*.*[v]<cr>
```

Display 3-3

When the A> prompt is displayed again, PIP is invoked to begin copying the transient commands from the original to the new work copy. The command line:

PIP B:=A:*.*[V]<cr>

tells CP/M to load the PIP utility from the diskette and copy all the files to disk drive B from disk drive A. The [V] is a PIP parameter which requests the utility to verify that the files being copied to drive B are exactly the same as the originals from drive A. As PIP copies each file, it will display the file name; when all designated files have been copied and verified, the A> prompt will be displayed. To test your new work copy, execute a cold and warm boot from it. You'll know you have created a working copy of CP/M when something similar to Display 1-1 is shown on the console. Make sure all the files were duplicated by requesting a file directory of the new copy.

It's a good practice to keep a work copy and a backup copy of your original CP/M diskette, and to use the original only to make additional copies.

For more detailed information about CP/M's SYSGEN and PIP commands, see Section 2.4.

4 CP/M AND APPLICATION PROGRAMS

An application program is one which performs a specific task for the user, such as managing a business ledger or providing word processing capabilities. A great number of application programs have been written for CP/M because of its ability to maintain disk files and its popularity among microcomputer manufacturers and users.

In this chapter we will briefly introduce you to some of the more common types of application programs available to you as a CP/M user. We also provide some basic guidelines for running application programs. Of course, this varies a great deal from program to program, but you will get an idea of what to expect when you begin to use application programs. For convenience, we sometimes refer to application programs as *applications*.

Types of Application Programs

Most of the applications written for CP/M are business programs. These include general ledger, accounts receivable/payable, payroll, order entry, inventory, and other types of programs which aid in the performance of tasks common to almost all businesses. Some applications are designed to meet the needs of a specific profession, such as medical/dental, legal, insurance, and accounting.

Word processing applications can be extremely useful in the business or home environment. A word processing application, such as MicroPro International's WordStar, has many advantages over CP/M's ED utility. Some of these advantages are described below:

- The ED utility is line-oriented. An efficient word processing application program is screen-oriented. As you enter text into a file, you see how it will appear on the page. It is easy to move forward or backward through the text, and know exactly where you are in the file by looking at the display. This greatly reduces the time spent searching for specific lines of text. Screen orientation also makes it easy to insert, delete, and move large blocks of text.

- When editing a file under the ED utility, you are restricted by the size of the edit buffer and your patience. If the entire file will not fit in the edit buffer at one time during the edit, you must write a portion of the file to a temporary file on disk, then append a number of lines from the source file to the buffer. This process can be time-consuming. A good word processing application will manipulate the flow of data for you.
- For the novice, word processing programs will often display during the editing session a list of command or function options available to the user. Such a list is called a *menu*. Display 4-1 shows how a menu for a word processing application program might appear on the screen. Below the menu is user input.

```
| Screen Functions          | Editing File:  CHAPTER4.CPM          |
| ------------------------------------------------------------- |
| Cursor:                                  | ^C  center        |
|    ^U  up        ^R  right word          | ^B  hold begin/end |
|    ^D  down      ^L  left word           | ^S  underscore begin/end |
| Scroll:                                  |                   |
|    ^A  line up   ^T  screen up           ------------------|
|    ^V  line down ^Y  screen down         | Other Menus:      |
| Delete:                                  |    ^P  Print      |
|    ^D  character ^W  word                |    ^H  Help       |
|    ^X  line                              |    ^F  File       |
-------------------------------------------------------------------

                        ^BCHAPTER FOUR
                  CP/M AND APPLICATION PROGRAMS ^B
An application program is one which performs a specific task for the user, such
as manage a business ledger or provide word processing capabilities. A great
number of []
```

Display 4-1

- Many word processing applications provide special capabilities, such as bold-print characters, underscore, automatic centering, and right-margin justification. These effects are difficult if not impossible to achieve using the ED utility. Their use can create excellent printed copy.

Running Application Programs

Always make a work copy of the original diskette containing your application program before you start using it. Follow the procedures described in Chapter Three. By doing this, you are assured of always

having an accurate copy of the software you purchased. When the recording media of the work copy diskette becomes worn from use, you can use the original to make a fresh work copy.

Application programs are usually accompanied by installation and operating instructions. Such instructions should explain the procedures for preparing the program to run on your computer system, and guide you in the proper use of the program's commands and utilities. The quality of the documentation accompanying the application program is generally a good indication of the program's usefulness. If you receive no documentation with your application program, the instructions may be on the disk itself. Always begin by reading the user instructions.

To run the application, first boot CP/M, then place the diskette that contains the application program in another disk drive. Obtain a directory of the application diskette using CP/M's DIR command. It is always wise to inspect the contents of the application diskette(s) to make sure they contain all the files required to run the program. The documentation you received should provide a list of the files included in the application. Check the directory of the diskette against this list.

In the rare case that you received no documentation with your software package, look for a file on the diskette with the extension .COM, .DOC, or .TXT. A .DOC (document) or .TXT (text) file should contain the information you need to install and operate the program. You can examine the contents of such a file using the TYPE command. To print a copy of the file as you view it on the screen, use the line editing command ^P, as shown in the example below:

```
A> TYPE PAYROLL.DOC ^P <cr>
```

You may find on your application diskette an installation program with a .COM file extension. In this case, you would type the name of the file, and CP/M would load the program into the TPA as it would one of its own transient programs. The program would then prompt you for the information it needed to adapt the program to your computer system. When the application is properly installed you can begin executing it.

Application programs written in a compiler language are sometimes delivered to the user in the form of .BAS (BASIC) files. You would be required, in this case, to compile the program before CP/M could control its operations. Instructions for compilation or modification should be included in the documentation you receive.

Since the application programs we're talking about operate under the control of CP/M, you'll be able to use some of CP/M's commands while the application is functioning. For example, the line editing command ^P (which causes displayed material to be simultaneously printed) can

be used whenever you want to retain a hard copy of your transactions. Another example is the ^C command; for many applications, this command causes a warmboot to be performed. Consult the user documentation to learn which of CP/M's commands can be utilized with your particular application program.

The best advice we can offer about selecting application programs is to ask a software vendor for comparisons and costs. Ask for a demonstration of the applications you are interested in. Then examine the documentation that accompanies the program. If it is informative, complete, and easy to understand, the program itself will probably also have these attributes. Software developers who take the time and effort to provide good instructions realize that it is the best representation of their products. Finally, the popularity of an application program is another good indication of its quality.

5 A SAMPLE SESSION

This chapter is meant to provide you with experience in using CP/M commands. The commands most often used for the creation and maintenance of files, and those issued to display and print file contents have been included in this sample session.

- If you have not done so, go back and read Chapter Two; you'll be using the DIR, ED, ERA, PIP, REN, STAT, SYSGEN, and TYPE commands, as well as some of the line editing commands. You'll save time by reading the descriptions of the commands before you use them here.
- You should not use your original end-user system diskette for this sample session. Load CP/M from the work copy you prepared (following the instructions in Chapter Three) into your computer from drive A.
- If your system came with a formatting utility, follow the instructions in Sections 3.1 and 3.2 to prepare another diskette for use during this session. If you don't have a formatting utility, be sure the diskette you will use came with the proper CP/M format for your particular computer system.

5.1 File Copy Transfer

After you've loaded the operating system from drive A, and CP/M's sign-on message and A> prompt have been displayed, place the new, empty diskette in drive B. Address drive B by typing:

B:<cr>

this will display the prompt for drive B, B>. Verify that the diskette in drive B is empty with the DIR command. Type:

DIR<cr>

If the diskette is empty, the screen of your computer should look like the one depicted in Display 5-1.

```
xx.K CP/M 2.2

A>B:<cr>
B>DIR<cr>

NO FILE

B>
```

Display 5-1

```
B>A:SYSGEN<cr>

SYSGEN VER 2.2

SOURCE DRIVE NAME (OR RETURN TO SKIP) A
SOURCE ON A, THEN TYPE RETURN <cr>

FUNCTION COMPLETE

DESTINATION DRIVE NAME (OR RETURN TO REBOOT) B
DESTINATION ON B, THEN TYPE RETURN <cr>

FUNCTION COMPLETE

DESTINATION DRIVE NAME (OR RETURN TO REBOOT) <cr>

B>
```

Display 5-2

If the diskette is not empty, DIR will have displayed the files contained on it. Erase any files on the diskette (or replace it with another if you want to keep the files), using the ERA command with a *.* ambiguous file reference.

Now use the SYSGEN utility to place the CP/M system on the empty diskette in drive B. Drive A will be the source drive, as shown in Display 5-2.

To place copies of the ED, PIP, and STAT transient commands on the diskette in drive B, use PIP from the system diskette in drive A. Do this by typing the command lines shown in Display 5-3.

```
B>A:PIP<cr>
*B:=A:ED.COM[V]<cr>
*B:=A:PIP.COM[V]<cr>
*B:=A:STAT.COM[V]<cr>
*^C

B>DIR<cr>

B: ED      COM : PIP      COM : STAT      COM

B>
```

Display 5-3

Now that the operating system, ED, PIP, and STAT have been copied onto the *session* diskette in drive B, go back to drive A by typing A: and remove the diskette from that drive. Remove the diskette from drive B, label it appropriately, place this session diskette in drive A, and type ^C to restart CP/M (this also lets the system know you have changed diskettes).

5.2 Creating a File

With ED on the session diskette, you can create a file and use it to see how other commands are employed with files. Our examples of file creation show how you can use ED to write letters, memos, or small reports and keep them stored as files. Files that are made up of alphanumeric symbols and punctuation marks are known as *text* files. These files are conventionally given an extension of TXT to identify them as containing readable characters.

```
A>ED SESSION.TXT<cr>

NEW FILE

 : *I<cr>
1: (Feel free to type anything you want,<cr>
2: perhaps something that will be of use to you.)<cr>
3: ^Z
 : *B#T<cr>
1: (Feel free to type anything you want,
2: perhaps something that will be of use to you.)
 : *E<cr>
```

Display 5-4

Invoke ED to create a file named SESSION.TXT by typing the command line shown in Display 5-4.

Because the file SESSION.TXT did not previously exist on the diskette, ED displays NEW FILE and creates two empty files on the diskette. This process is described in Section 2.4, page 31.

Now you may begin issuing commands to ED. The first command you'll want to enter is I<cr> . As discussed in our description of ED in Chapter Two, this allows you to begin inserting text that will eventually be placed in the SESSION.TXT file.

If you have read our description of the ED utility, you know about the command options available to you. Feel free to experiment with them now (we don't offer you line-by-line text copy to insert in the file). Type in what you like, perhaps something that will be of use to you.

You should limit the length of your lines so that they don't go off the screen. When you've typed in all the text you want, enter a ^Z to signal ED that you are finished with insert mode.

When ED's * prompt is redisplayed, type: B#T<cr> . These two commands tell ED to go to the Beginning of the text buffer and Type (display) all the lines of your file.

Now enter E<cr> to End the EDit session.

Find out how large your file is by using STAT:

A> STAT SESSION.TXT<cr>

You may want to rename SESSION.TXT. You can do this and still retain the original file by using PIP:

```
A> PIP newname.ext=
   SESSION.TXT[EV]^P<cr>
```

This single command line combines the function of the REN command, the file copy transfer capability of PIP, and the TYPE command. If you include the ^P line editing command, make sure the printer is ready.

To protect your new file you can set it to Read/Only status with STAT:

```
A> STAT newname.ext $R/O<cr>
```

Be aware, however, that any attempt to EDit the file will result in an error message. You can change it back to Read/Write status by typing:

```
A> STAT newname.ext $R/W<cr>
```

With your file protected, you can now experiment with the other CP/M commands you have read about in this guide.

APPENDIX

A COMMAND/FUNCTION REFERENCE

The Command/Function Table provides the following information for all CP/M commands: command type (shown as B-I for built-in, L-E for line editing, and TRAN for transient); a brief functional description; and a page number indicating where detailed information on the functions and variations of the command can be found in Chapter Two. For quick references to individual command structures, see the Command Reference Card at the end of this book. Those commands not fully supported by this manual are enclosed in brackets.

COMMAND	TYPE	FUNCTION	PAGE
[ASM]	**TRAN**	Loads and runs the CP/M assembly program against a specified disk file.	66
^C	**L-E**	Executes a warmboot.	17
[DDT]	**TRAN**	CP/M's Dynamic Debugging Tool.	66
DIR	**B-I**	Causes a directory of the files contained on a disk to be displayed on the console. Disk drive and file name may be specified; file names may be ambiguous or unambiguous.	21
^E	**L-E**	Moves cursor to the beginning of the next line but does not issue the command being typed.	18
ED	**TRAN**	Creates and alters text files.	33
ERA	**B-I**	Erases files from a disk. A disk drive and file name may be specified; the file name may be ambiguous or unambiguous.	23
^H	**L-E**	Deletes and erases the last character of a command line.	19

^J	**L-E**	Issues the command line and moves the cursor to the beginning of the next line. (Same as a carriage return.)	19
[LOAD]	**TRAN**	Converts hexadecimal files into .COM files.	67
^M	**L-E**	Same as ^J.	19
[MOVCPM]	**TRAN**	Permits a systems programmer to configure the CP/M system so that it may be used in particular memory allo-ments.	67
^P	**L-E**	Toggles to enable or disable screen output to be simultaneously sent to the LST: device (usually a printer).	19
PIP	**TRAN**	Gives you the ability to move files from one device to another.	50
^Q	**L-E**	Resumes scrolling after it has been halted.	19
^R	**L-E**	Retypes (displays) the command on the next line.	19
REN	**B-E**	Allows you to rename a file. A disk drive may be specified, but the drive for the old and new names must be the same.	24
^S	**L-E**	Stops the display of console screen output.	20
[SAVE]	**B-I**	Saves the contents of the Transient Program Area (TPA) in a user-specified diskette file.	24
STAT	**TRAN**	Displays the amount of available memory remaining on a disk and the size of particular files. STAT also allows you to display/alter current system device assignments and set file attributes.	57
SUBMIT	**TRAN**	Submits for execution a group of commands which have been previously written in a .SUB file.	64
SYSGEN	**TRAN**	Copies the CP/M system tracks from one disk to another.	64

TYPE	**B-I**	Causes the contents of a text file to be displayed on the console device. The name and type of the file must be specified (unambiguous). A disk drive may be specified. Stop and resume scrolling of the file being displayed by typing ^S.	25
^U	**L-E**	Moves the cursor to the beginning of the next line and cancels, without removing from the screen, the command line.	20
USER	**B-I**	Changes the currently logged user number.	26
^X	**L-E**	Cancels and erases the command line. Returns the cursor to the beginning of the current line behind the A> prompt.	20
[XSUB]	**TRAN**	Permits console input to be accepted as part of the commands being executed in a .SUB file by SUBMIT.	67

APPENDIX

B GLOSSARY

This glossary is provided to aid in your understanding of some terms frequently used in conjunction with CP/M and the microcomputer environment. It defines these terms as they are used in this document. In most cases, the definitions are fairly standard, but you may encounter some variation in other sources. We have included synonyms for terms wherever applicable.

ambiguous/unambiguous file references

An unambiguous file reference occurs in a command line when a specific file name and extension are designated. For example, in the command line STAT PIP.COM <cr>, PIP.COM is an unambiguous file reference. Two symbols are used in the formation of ambiguous file references: the asterisk (*), which may replace an entire file name or extension; and the question mark (?), which replaces a single character. The following are examples of command lines in which ambiguous file references occur:

DIR *.COM <cr> This command will cause a directory of files with any file name and the extension .COM to be displayed.

DIR YEAR82.MO? <cr>
This command displays a directory of files with the file name YEAR82 and an extension beginning with the letters MO and any or no final character.

(See Section 1.6 "General Command Structure" for further information on ambiguous and unambiguous file references.)

application program

Programs which perform a specific job for the user, such as create and maintain a customer mailing list or an accounts receivable ledger. There are a multitude of applications available for use with CP/M. (See Chapter Four "CP/M and Application Programs.")

ASCII (American Standard Code for Information Interchange)

A character set consisting of alphanumeric, punctuation, and

control characters used in the transmission of information in an electronic environment, such as your computer.

BDOS (Basic Disk Operating System)
The segment of the operating system which directs all activities associated with the disk drives, console I/O, printer output, and reader/punch I/O. (See Section 2.1 "An Overview of CP/M's Internal Structure.")

BIOS (Basic Input/Output System)
The segment of the operating system which performs all physical operations, especially disk I/O operations under the direction of BDOS. BIOS must be tailored or configured to operate correctly with a particular computer system. (See Section 2.1 "An Overview of CP/M's Internal Structure.")

bootstrapping
Bootstrapping, bootup, coldboot and loading, used interchangeably, describe the process of starting the CP/M program for the first time in a session. That is, after power has been applied to the microcomputer, CP/M is coldbooted by placing the CP/M diskette in the disk drive and pressing the Reset key (if your computer is equipped with one). Consult the manual that came with your computer for more specific bootstrapping procedures.

buffer
(See **text buffer**)

byte
A unit of memory capable of storing one character. 1,024 bytes are called a kilobyte. The storage capacity of a microcomputer or diskette is always expressed in bytes, usually kilobytes (for example, 64KB or 161K bytes).

CCP (Console Command Processor)
The segment of the operating system that accepts user input from the keyboard. The CCP contains and executes the built-in and transient commands. (See Section 2.1 "An Overview of CP/M's Internal Structure.")

coldboot/start
(See **bootstrapping**)

command, command line
A specific instruction issued by the user to CP/M. Commands are generally issued by typing the name of the command, followed by any required parameters, and then depressing the Return or Enter key. (See Section 1.6 "General Command Structure.")

command parameters
(See **parameters**)

console
The components of your computer system which comprise the keyboard and display. The term may at times be applied individually to the keyboard or display.

control character
A character which is formed by pressing and holding down the CTRL or Control key while typing an alphabetical key (or certain punctuation keys) at the same time. A control character may initiate, modify, or end command operation. Input of a control character may or may not appear as a graphic character on the screen. CP/M's line editing commands are control characters. Control characters which are not line editing commands are usually displayed on the console as ^x where x is the alpha or punctuation key pressed. Certain ED commands are also control characters.

CPU (Central Processing Unit)
The component in your computer that interprets and executes program instructions. The CP/M operating system is designed to run on the Intel 8080 and 8085 CPUs and the Zilog Z80 CPU.

CRT (Cathode Ray Tube)
An electronic vacuum tube, similar to a TV tube, which is used to display user input and computer output. Also called console, display, or screen.

cursor, cursor position
The location marker on the console screen that indicates where the next character input from the keyboard will appear.

data
Information composed of numbers, letters, and/or symbols that are processed by the computer.

default
In a command or computer function, when a certain value, option, or attribute is not designated, the computer or CP/M will assume a predetermined value, called a default value. For example, when no disk drive is specified in a command, CP/M will *default* to the currently logged drive.

device
An individual component of a computer (such as a printer) which receives or outputs data in order to perform a particular function.

CP/M supports a set of logical devices, each of which may correspond alternately to various physical devices. (For more information concerning logical to physical device assignments, see the STAT command description, "Device Assignment," in Section 2.4.)

directory, disk directory

A catalog of the files resident on a diskette. The directory is maintained and modified by CP/M on the diskette so that it can manage file input/output and allocation operations. A file may be composed of a number of extents and will have a corresponding number of directory entries. However, only the first directory entry of each file will be displayed in response to a DIR command.

diskette

A flexible magnetic disk encased in a thin protective cover. Information is stored in files on a diskette and may be accessed by the CP/M operating system. "Floppies" come in two sizes (8-inch and 5.25-inch), but there are many other variables. For example, diskettes may be single- or double-sided, single- or double-density, soft- or hard-sectored, and the combinations are almost endless. Consult the vendor from whom you purchased your computer for the correct diskettes for your system. Also called: disk, floppy, floppy diskette.

display

(See **CRT**)

edit buffer

(See **text buffer**)

error condition, message

Any state or occurrence which prohibits, interrupts, or cancels computer operation is an error condition. Such conditions are often accompanied by an error message (displayed on the console) identifying the problem. (See Section 2.5 "General Error Messages and Conditions.")

extension, file extension

(See **file**)

extent

An organizational unit of a predetermined number of file records. The number of records in an extent depends upon the particular CP/M implementation. Each extent of a file accounts for one disk directory entry.

file, disk file

Any collection of related data stored by CP/M as a separate unit on a diskette is a file. Every file has a unique name of 1 to 8 characters and may have an extension of 1 to 3 characters. The extension, or type, is separated from the file name by a period. (See Sections 1.1 and 1.2, "Data Storage and File Structure," and "File Names and Extensions," respectively.)

floppy, floppy diskette

(See **diskette**)

hexadecimal, hex

A number system that is based on powers of 16, instead of powers of 10, as is the decimal system. Where the decimal system uses the digits from 0 to 9, the hex system uses digits from 0 to F, adding A, B, C, D, E, and F to signify the six extra numerals having decimal values of 10 through 15. Hex numbers are often identified, as in these examples: FF08 hex, E9H.

interface

Generally, any shared boundary or medium of communication, as between computer components, between software and hardware, or between you, the user, and your computer.

I/O (Input/Output)

Input is the submission of data, as to the computer by the user, usually through the keyboard. Output is data transferred by the computer to a device such as a printer or CRT for access by the user. Devices which either accept user input or produce computer output, or both, are called I/O devices.

IOBYTE

A byte of memory reserved by CP/M to record the current physical to logical device assignments. (See the STAT command description, "Device Assignment," in Section 2.4.)

I/O device

(See **device**)

kilobyte

(See **byte**)

logged disk, drive

The disk drive from which CP/M is presently processing commands. The logged disk is changed by typing: x:<cr> where x is the drive to be logged in.

memory

The part of a computer system used to store data. Memory may consist of hardware components within the computer (i.e., RAM or ROM) and mass storage media, such as diskettes or magnetic tape, outside the computer.

mnemonic

An abbreviation of one or several words so that the abbreviation helps you to remember what it stands for. Examples of mnemonics used in conjunction with CP/M are: TPA for Transient Program Area; PIP for Peripheral Interchange Program; and DIR for DIRectory. You should use a mnemonic whenever you name a file.

operating system

A group of related programs which collectively manage all the input/output and resource allocation functions of a computer. Operating systems, such as CP/M, receive instructions from the user to store and retrieve information, move files from one location to another, inquire into the status of a diskette or device, and execute outside programs. (See the introductory paragraphs of Chapter One, "About CP/M," for more information.)

parameters, command parameters

Auxiliary portions of a command line (such as file names and extensions, disk drive designations, or special instructions in the form of one or two characters) which modify the execution of the command. The following example shows a PIP command line that includes several parameters:

```
A> PIP B:=A:*.COM[V]<cr>
```

In this case, the Verify (V) parameter must be enclosed within brackets. The command instructs CP/M to copy (PIP) to drive B from drive A all files with the extension .COM. The V parameter tells CP/M to verify that the copied files exactly resemble the original files.

peripheral device/equipment

Components of a computer system other than the CPU which provide extra I/O or communication capabilities.

physical/logical

Examples of a microcomputer's physical units or devices are the keyboard, screen, printer, disk drives, etc. CP/M, however, when performing the input/output operations of the system, ignores the physical unit designations and considers them to be logical units. The advantage is that CP/M can operate in a wider variety of computer configurations; for example, output is sent to a (logical)

"listing device" which the operating system equates to *either* a printer *or* a punch device. (See the STAT command description, "Device Assignment," in Section 2.4.)

physical/logical device
(See **device**)

prompt
When CP/M or one of its utilities is prepared to accept an instruction from the user, it will display a *prompt* on the left side of the screen. CP/M's prompt is composed of the letter of the currently logged drive and a "greater than" symbol (A>); the ED utility uses a colon, space, and asterisk (: *) for its prompt; and the PIP utility uses an asterisk (*).

RAM (Random Access Memory)
The portion of the computer's memory where data may be stored and easily altered or erased. RAM is where CP/M resides when it is loaded into your computer.

record
A unit of disk memory consisting of 128 bytes. CP/M uses records as the basic building blocks in the construction of files. (See Section 1.1 "Data Storage and File Structure.")

R/O (Read Only)
An attribute applied to a file, disk, or disk drive which prevents data from being written to it. When the R/O attribute is applied through the STAT command to a disk drive, it is only effective until the next warmboot or coldboot. When applied to a file, it remains in effect until changed to Read/Write.

ROM (Read Only Memory)
The portion of the computer's memory whose contents remain constant (i.e., not meant to be altered or erased). ROM often contains programs that are essential to the operation of the computer system.

R/W (Read/Write)
The default attribute applied to a file, disk, or disk drive which allows data to be written to it.

screen
(See **CRT**)

scrolling
The continuous output of data to the CRT, so that new data appears at the bottom of the screen, while old data disappears from the top of the screen.

sectors
In the format of a diskette, equal portions of a track, usually 128, 256, or 1024 bytes long. (See Section 1.1 "Data Storage and File Structure.")

signon message
The message that appears on the CRT when the operating system is cold started. (An example of the CP/M signon message can be found in Display 1-1.)

text buffer
A portion of your computer's memory reserved by the ED utility for text that is being edited under its control. Also called *edit buffer*.

TPA (Transient Program Area)
The part of the computer's memory that CP/M reserves for the execution of transient commands and other programs. (See Section 2.1 "CP/M's Internal Structure.")

tracks
A series of concentric bands on the recording surface of a diskette. (See Section 1.1 "Data Storage and File Structure.")

unambiguous file references
(See **ambiguous/unambiguous file references**)

user area
A separate logical area of a diskette composed of files with a common user area number. (The description of the USER command in Section 2.3 tells how to access and activate user areas.)

utility
A program that performs the ordinary tasks of an operating system, such as transferring files from one location to another or reporting file statistics. CP/M's transient commands are all utilities.

warmboot/start
The process of reloading the CCP and BDOS portions of CP/M after running a program or utility, or upon action of the user. This is often necessary when the user requests a directory of a diskette that has just replaced another diskette for which a directory was previously requested. The same is true for a similar situation using the STAT command. A warmboot is performed simply by typing ^C.

write-protect/enable
A procedure that prevents or enables data to be recorded on a

diskette by either placing an adhesive tab over a notch on the diskette sleeve or leaving the notch open.

5.25-inch diskette:

Notch open—write-enabled
Notch closed—write-protected

8-inch diskette:

Notch open—write-protected
Notch closed—write-enabled

(See Section 1.3 "Diskette Care.")

APPENDIX

C REFERENCES

There is an extensive body of documentation devoted to CP/M and specific aspects of its operation. Digital Research, Inc., of course, provides many informative manuals, and there are a number of tutorials, as well as publications for more advanced CP/M users available on the open market. Here are a few you may be interested in looking at:

Digital Research Documentation

An Introduction to CP/M Features and Facilities

CP/M 2.2 User's Guide

CP/M Assembler (ASM) User's Guide

CP/M Dynamic Debugging Tool (DDT)

ED: A Context Editor for the CP/M Disk System User's Manual

CP/M 2.2 Interface Guide

CP/M 2.2 Alteration Guide

Tutorials

CP/M User Guide, by Thom Hogan
Osborne/McGraw-Hill

CP/M Primer, by Stephen M. Murtha and Mitchell Waite
Howard W. Sams and Co., Inc.

Using CP/M, by Judi Fernandez and Ruth Ashley
John Wiley and Sons

How to Get Started with CP/M, by C. Townsend
Dilithium Press

INDEX

CP/M COMMAND REFERENCE CARD

The following symbols are used on this card:

x:	designated disk drive; when omitted, command defaults to currently logged drive
name	file name
ext	file extension, type
dev	system device
^	Control key; held down while next character is typed
<cr>	carriage return

BUILT-IN COMMANDS

DIR x: Displays a directory of files on a disk.

DIR x:name.ext
Displays a directory of all files with the specified name and extention; ambiguous or unambiguous.

ERA x:name.ext
Erases a file from the disk; ambiguous or unambiguous.

REN x:newname.ext = oldname.ext
Renames a file on a disk.

SAVE nn x:name.ext
Saves nn pages of TPA in a disk file.

TYPE x:name.ext
Displays contents of a file on CRT screen.

USER n Specifies a user area number (0–15, incl.)

TRANSIENT COMMANDS

ASM Assembles an .ASM source file.

DDT x:name.ext
Alters and tests a program.

DUMP x:name.ext
Displays the hexadecimal representations of each byte stored in a file.

ED x:name.ext
Creates or edits a file.

ED COMMANDS

nA	Appends n lines from the original file to the edit buffer.
B/-B	Moves the CP to the start or end of the edit buffer.
nC/-nC	Moves the CP n character positions forward or backward.
nD/-nD	Deletes n characters after or before the CP.
E	Ends the edit session.
nFstrg^Z	Finds the nth occurrence of the string "strg" in the edit buffer after the CP.
H	Saves file and moves to the beginning of the edit buffer.
I	Enters the insert mode.
Istrg^Z	Inserts the string "strg" in the edit buffer after the CP.
Istrg	Inserts the string "strg" and a <cr> in the edit buffer after the CP.
nJfindstrg^Zinsertstrg^Zendstrg^Z	After the CP, find "findstrg" and insert "insertstrg" after it, deleting all following characters up to but not including "endstrg." Repeats n times.
nK/-nK	Deletes n lines after or before the CP.
nL/-nL	Moves the CP n lines forward or backward.
nMcommandstrg^Z	Executes the ED commands in "commandstrg" n times
nNstrg^Z	Finds the nth occurrence of the string "strg" in the edit buffer and the original file.
O	Erases the edit buffer and returns to the beginning of the original file.
nP/-nP	Moves the CP forward or backward one page and displays the following page. Repeats for n pages.
Q	Ends the edit session. The original file remains unaltered.
R	Reads and copies the block move file from the disk and inserts it into the edit buffer after the CP.
Rname	Reads and copies the file name with extension .LIB from the disk into the edit buffer after the CP.
nSfindstrg^Zreplacestrg^Z	After CP, finds "findstrg" and replaces it with "replacestrg." Repeats n times.
nT/-nT	Displays n lines after or before the CP.
U/-U	Converts alphabetic input from lower case to upper. U starts, -U ends conversion.
V	Displays number of free bytes and total size of the edit buffer.
V/-V	Displays (V) or suppresses (-V) line numbers.
nW	Writes first n lines from the edit buffer to the temporary file. Deletes first n lines from the edit buffer.
nX	Copies n lines after the CP from the edit buffer to the temporary block move file.
nZ	Delays the execution of the next command.
n:	Moves the CP to the start of line n.
:m	Executes the specified command through line m.
n/-n	Moves forward or backward n lines and displays a line.
LOAD name	Converts a .HEX file to a .COM file.
MOVCPM mm	Creates an mmK version of CP/M and executes it, but does not store it on disk.
MOVCPM mm *	Creates an mmK version of CP/M and leaves it in memory to be transferred to disk.
MOVCPM * *	Creates a maximum memory version of CP/M and leaves it in memory to be transferred to disk.
PIP	Loads the PIP utility into memory and displays the PIP prompt (*).

PIP x:name.ext = x:name.ext[p]
Copies a file.

PIP x:newname.ext = x:oldname,x:oldname
Creates a new file from two old files.

PIP x:name.ext = dev:[p]
Copies data from a PIP source device to a file.

PIP dev: = x:name.ext[p]
Copies data from a file to a device.

PIP dev: = dev:[p]
Copies data from one device to another.

PIP PARAMETERS

B	Specifies block mode transfer.
Dn	Deletes characters after the nth column.
E	Displays the contents of a file as it is being copied.
F	Removes form feeds from the original file.
Gn	Copies files from user area n.
H	Specifies transfer to be in Hex format.
I	Hex format. (Intel)
L	Converts uppercase to lowercase letters.
N	Adds numbers to each line of the copied file.
O	Transfers non-ASCII file.
Pn	Inserts a form feed after each n line.
Qs^z	Copies from the first character to string s.
R	Copies a system file.
Ss^Z	Copies from string s to the end of the file.
Tn	Sets tab stops at every nth column.
U	Converts lowercase to uppercase letters.
V	Verifies that original and copied files are the same.
W	Copies into an R/O file.
Z	Sets parity bit to zero.

STAT Displays amount of free space on drives accessed since last system boot.

STAT x: Displays the amount of free space on disk x:.

STAT x:name.ext
Displays the attributes and size of a file.

STAT x:name.ext $atr
Assigns the attribute atr to a file(s).

STAT Attributes:

R/O	Read-Only file
R/W	Read-Write file
SYS	SYStem file
DIR	DIRectory file

STAT DEV:
Displays the current physical to logical device assignments.

STAT USR:
Displays the current user number and/or user numbers of all files on active disk.

STAT x:DSK:
Displays disk parameter information for a disk drive.

STAT x: = R/O
Assigns a temporary R/O status to a disk.

STAT VAL:
Displays possible physical to logical device assignments. Displays a partial STAT command summary.

STAT logdev: = phydev:
Assigns a physical to a logical device.

Possible Physical to Logical Device Assignments:

CON: =	TTY:,	CRT:,	BAT:,	UC1:
RDR: =	TTY:,	PTR:,	UR1:,	UR2:
PUN: =	TTY:,	PTP:,	UP1:,	UP2:
LST: =	TTY:,	CRT:,	LPT:,	UL1:

SUBMIT name
Creates a $$$.SUB file that contains the CP/M commands in name.SUB and then executes the commands from $$$.SUB.

SUBMIT name p
Creates a $$$.SUB file that contains partially completed or complete CP/M commands in name.SUB. CP/M completes commands with the specified parameters and then executes the commands from $$$.SUB.

SYSGEN Copies the CP/M operating system.

XSUB Allows input to programs executed in a .SUB file.

LINE EDITING COMMANDS

^C	Activates a CP/M warm start.
^E	Moves cursor to beginning of next line.
^H	Erases a character from the end of command line. (Same as Backspace key.)
^J or ^M	Performs a carriage return. (Same as Return, Enter, Line Feed keys.)
^P	Turns the LST: device on/off.
^Q	Resumes scrolling on screen.
^R	Displays corrected command line.
^S	Stops scrolling on screen.
^U	Cancels but does not erase the command line.
^X	Cancels and erases the command line.

FILE NAMING CONVENTIONS

File Names:
Consist of 1 to 8 characters.

File Extensions or Types:
Consist of 0 to 3 characters separated from file name by a period.

Unambiguous and Ambiguous File References:
An unambiguous file reference includes a specified file name and extension. Ambiguous file references include an asterisk (*) and/or question mark(s) (?) in the file name or extension.

Invalid Characters:
< > . , : ; = ? * [ ]

5208-1

GETTING STARTED WITH CP/M®

ROB PATTEN & PAUL CALANDRINO

Developed for the novice computer user who is looking for a gentle, step-by-step initiation into using the world's most popular microcomputer operating system. From the simplest explanation of what CP/M is and what an operating system does, to a detailed walk-through of a productive working session, *Getting Started with CP/M®* concentrates on answering beginners' questions: How do I load the program? How do I name files? How do I edit programs, care for my diskettes, use built-in CP/M commands, or run applications programs under CP/M? The answers are here, along with many others.

Getting Started with CP/M® also offers a concise summary of all operating system commands, a glossary of microcomputing, a valuable bibliography, and a handy pull-out Command Reference Card to keep by your machine.

OTHER BOOKS of INTEREST...

CP/M® REVEALED
Jack D. Dennon

Explore and master the more powerful features of CP/M. The author clearly explains the technical aspects of the console command processor (CPP — CP/M's line of communication with the user), the system manager (BDOS), and the input/output manager (BIOS).

Advanced users will be able to use CP/M's built-in editor and assembler to control these powerful systems directly, to change memory sizes, map disk space, and take advantage of the full potential of CP/M's input/output utilities. Less advanced users can start with early chapters explaining how to log into the system, manage disk files, and write and run programs using CP/M's higher level commands and the programming languages that go along with them. #5204-9, paper, 192 pages.

Z-80 AND 8080 ASSEMBLY LANGUAGE PROGRAMMING
Kathe Spracklen

An excellent introduction to assembly language programming in CP/M's native tongue — written by the author of *Sargon*, one of the most advanced chess-playing programs ever written for a microcomputer. The author emphasizes practical application software development throughout the book, using many diagrams and examples to illustrate creative approaches to assembly language programming. Exercises accompany each chapter, and all answers are contained in an appendix. #5167-0, paper, 176 pages.

HAYDEN BOOK COMPANY, INC.
Rochelle Park, New Jersey

ISBN 0-8104-5208-1