MS-DOS 2.1 Reference
For the PlusPC

Rev. A

First VICTOR printing June, 1985.

ISBN 0-88182-143-8 Printed in U.S.A.

Important Software Diskette Information

For your own protection, do not use this product until you have made a backup copy of your software diskette(s). The backup procedure is described in the user's guide for your computer.

Please read the DISKID file on your new software diskette. DISKID contains important information including:

- The part number of the diskette assembly.
- The software library disk number (for internal use only).
- The date of the DISKID file.
- A list of files on the diskette, with version number, date, and description for each one.
- Configuration information (when applicable).
- Notes giving special instructions for using the product.
- Information not contained in the current manual, including updates, any known bugs, additions, and deletions.

To read the DISKID file onscreen, follow these steps:

1. Load the operating system.
2. Remove your system diskette and insert your new software diskette.
3. Enter —

   ```
   type diskid(cr)
   ```

4. The contents of the DISKID file is displayed on the screen. If the file is large (more than 24 lines), the screen display will scroll. Type CTRL-S to freeze the screen display; type CTRL-S again to continue scrolling.

Preface

This manual contains these seven chapters and four appendixes:

- Chapter 1 discusses the structure of MS-DOS, how the system uses memory and disk space, how to load MS-DOS, and other introductory topics.
- Chapter 2 describes the types of disk files: system files and utilities, data/text files, and batch files. This chapter also describes how to use device files, and gives the rules for naming files and for using wildcard characters.
- Chapter 3 discusses the hierarchical (tree-structured) MS-DOS directory system, and tells how to use subdirectory names, create and delete directories, move between directories, and find files in the directory system.
- Chapter 4 explains how to use EDLIN, a text editor used mainly to edit source files.
- Chapter 5 tells how to edit MS-DOS command lines, how to redirect input/output, and how to "filter" and "pipe" data.
- Chapter 6 describes all the MS-DOS commands—the internal, external, and batch commands, and the system configuration commands. This chapter gives the syntax for each command, explains parameters used with each command, and gives examples of usage.
- Chapter 7 lists and describes messages the system might display during bootup and during normal operations.
- Appendix A is an ASCII to hexadecimal conversion table.
- Appendix B lists the device input/output errors.
- Appendix C describes the ANSI escape sequences for IBM mode.
- Appendix D contains information on the DEBUG program. All the DEBUG commands are listed alphabetically with a description.

Manual Conventions

This manual uses the following conventions:

- When commands are explained for the first time, the command format is set off in boldface, such as:

  DO filename /switch(cr)

  —The parts of the command that you type exactly as shown (such as the command name) are uppercase.

  —Parts of the command that you vary or omit to fit the situation (such as filenames and drive names) are lowercase.

  —When you type a command, you can use upper- or lowercase for any command elements.

- In examples and command formats, pressing the Return key is shown as (cr), and typing a space is shown as (sp).
- Some examples show the screen, including what you type and the system's response. In these screen examples, what you type is underlined and lowercase. For example:

```
A>print myfile(cr)
```

- In the text, names of commands, files, and programs appear in uppercase. Examples are COPY and TEXT.DOC.
- The CTRL key acts as the "Control" key on your computer. In text, "CTRL-" represents the CTRL key. For example, CTRL-C represents typing the control function of key C. In screen examples, however, the CTRL key is shown as a caret (^), such as ^C.

Manual Conventions

This manual uses the following conventions.

- When commands are presented for the first time, the command format is in a boldface [illegible]:

  GO filename [keyword]

  — The parts of the command that you type exactly as shown (such as the command name) are [illegible].

  — Parts of the command that you supply or omit to fit the situation (such as filenames and drive names) are lowercase.

  — When you type a command, you can use upper- or lowercase for any command element.

- In examples and command formats, pressing the Return key is shown as [illegible] and a space is shown as [illegible].

- Some examples show the screen, indicating what you type and the system's response. In these screen examples, what you type is underlined and lowercase. For example:

```
A>[illegible]
```

- In the text, names of commands, files, and programs appear in uppercase. Examples are [illegible] and PRINT.EXE.

- The CTRL key acts as a control key on your computer. In text, "CTRL-" represents the CTRL key. For example, CTRL-C represents typing [illegible] key C. In screen examples, however, the CTRL key is shown as a caret (^), such as ^C.

Contents

APPENDIXES

Chapters

Appendixes

A

B

C

D

Introduction

MS-DOS (Disk Operating System) is a group of programs that:

- ▶ Manage files and directories
- ▶ Process commands
- ▶ Control application programs designed for MS-DOS
- ▶ Control the operation of your keyboard, monitor, disk drive(s), and printer(s)

The first part of this chapter gives short descriptions of the system files and utilities that make up MS-DOS. There is a brief explanation of how the PlusPC stores the system and program files on the system diskette. Also included is a discussion of the way MS-DOS uses memory (RAM) and disk space. The last part of the chapter describes system configuration and the procedures for loading MS-DOS and for entering the date and time.

1.1 MS-DOS System Structure

The PlusPC contains three system files/programs. There are two versions of each file: one for I mode, and one for V mode. Each version is prefaced with an "I" for IBM, or a "V" for VICTOR. MS-DOS consists of the following files/programs:

▶ System Files

ICOMMAND.COM/VCOMMAND.COM—the command processor that accepts commands from the keyboard and runs the programs that process these commands.

ICONFIG.SYS/VCONFIG.SYS—system initialization routines and commands for changing the system configuration (see Chapter 6.3).

IDOS.SYS/VDOS.SYS—the operating system programs that manage files and application programs. These system files are "hidden"; that is, they do not appear in directory listings.

▶ PlusPC External Commands

ASSIGN assigns a logical designator to a drive.

CHKDSK displays the amount of disk space occupied by files and tells you how much space is available for storage. CHKDSK also scans the directory for errors and can correct allocation problems.

COMMAND loads the command processor (ICOMMAND.COM/VCOMMAND.COM) into memory.

COMP compares the contents of one file to the contents of another file.

CONCAT concatenates (combines) files, displays the files onscreen, or sends them to a file.

DISKCOMP compares the files of two diskettes.

DISKCOPY copies the contents of one diskette onto another diskette. DISKCOPY works in V mode only.

EDLIN is the MS-DOS line editor program.

EXE2BIN converts executable files (.EXE) to the binary format (.COM).

EXESIZE sets a specified file's .EXE header values.

FGREP searches through files or directories for a specified expression, and then prints the names of the files and the lines containing the expression.

FILCOM (file comparison utility) compares two files and displays the differences between the two.

FIND searches for a specified string of text.

FORMAT prepares a diskette to receive MS-DOS files. FORMAT formats diskettes in VICTOR format and works in V mode only.

GRAPHICS sets up a graphics printer to allow printing of graphics display screens. You can use GRAPHICS in I mode only.

LS alphabetically lists a directory, or lists selected parts of a directory according to specified constraints.

MODCON changes the “soft console” (keyboard and character set files) used by your computer, in V mode only.

MODE sets up printer ports and asynchronous communications adapters. There is one MODE command, but some of the forms of this command do not pertain to V mode.

MORE displays the contents of files one screen at a time.

MV moves files (makes a copy of a file, and deletes the original), and renames files.

PRINT queues a list of files for printing.

RECOVER recovers lost data from a file or from a whole disk.

RETROSYS copies the system files to VICTOR format diskettes or to the fixed disk.

SDCOPY copies the contents of one diskette onto another diskette. SDCOPY requires only one floppy drive. You can use SDCOPY in V mode only, and you can use only VICTOR format diskettes.

SEARCH selects files from a given set of files according to matching constraints, and performs a specified action on the matching files.

SORT sorts specific lines of data from a file or arranges the file in ascending or descending order.

TAIL displays the final lines of a file.

TREE lists all directory paths on a drive.

WC counts words and/or lines in a file or a group of files.

Note: Chapter 6.4 contains detailed descriptions of all the MS-DOS commands alphabetically. Refer to Chapter 6.4 for explanations of commands as you read the earlier chapters.

The hidden system file IDOS.SYS/VDOS.SYS contains programs that execute automatically when MS-DOS loads into memory. IDOS.SYS/ VDOS.SYS and ICOMMAND.COM/VCOMMAND.COM control the system resources. The relationship between these files and system resources is shown in Figure 1-1. The relationship of the two files remains the same in either mode.

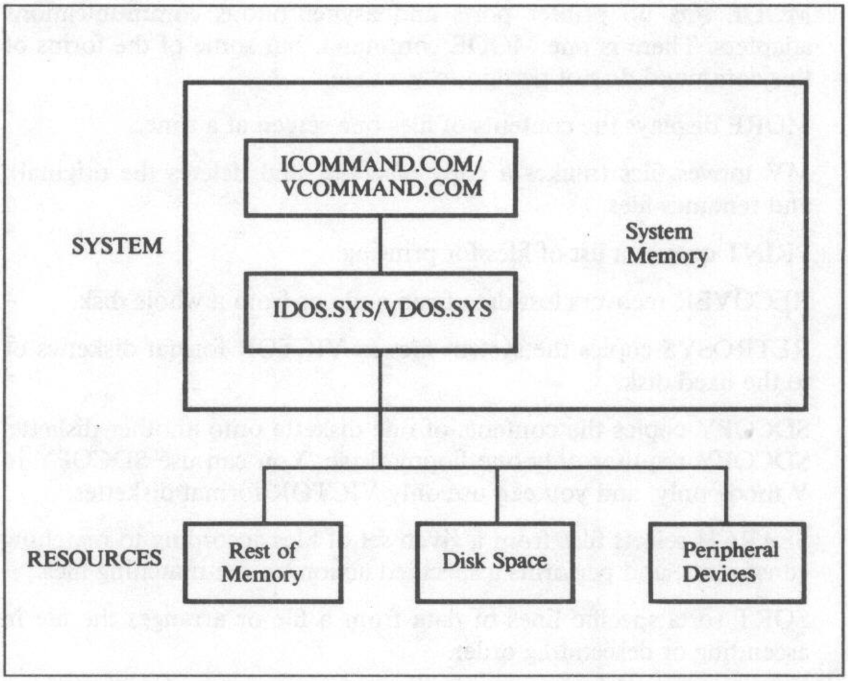


Figure 1-1: MS-DOS and Its Resources

The system's resources are its internal memory (RAM) and external storage (disk space), plus connections to the CRT, keyboard, printer(s), other peripheral devices, and a network configuration, if any.

Memory and the MS-DOS file system can be thought of as the external organization of system resources. MS-DOS supports "device-independent I/O." This means that you can treat devices (such as your printer, keyboard, and CRT) as if they are files when you do input/output processing (see Chapter 2.2). The system's memory and disk space are described in Chapters 1.1.2 and 1.1.3.

1.1.1 The PlusPC System Diskette

On your PlusPC system diskette, VICTOR has provided a subdirectory structure for storing the MS-DOS system and program files. The program files for the PlusPC MS-DOS commands are contained in three different directories called libraries:

- ▶ VLIB is the V mode library. The V mode library contains the program files that are specific to the V mode of your PlusPC system.
- ▶ ILIB is the I mode library. The I mode library contains the program files that are specific to the I mode of your PlusPC system.
- ▶ COMNLIB is the common library. The common library contains the program files that are common, or shared by both the V and I modes of your PlusPC system.

See the *PlusPC User's Guide* for a list of the program files contained in each library. In Chapter 6.4 of this manual, all the MS-DOS commands have special notes if any of their features or parameters are specific to either I mode or V mode.

CAUTION: It is important that you maintain the subdirectory structure provided for you on the system diskette. A path has been set up on the diskette so that the correct version of a program is executed for the mode you are using. If you move any of the MS-DOS program files from the subdirectories in which they are located when you receive the system diskette, you might use the incorrect program. If this happens you can overwrite or reformat existing data. By maintaining the MS-DOS subdirectory structure as it is already set up, you are ensured of using the correct version of each command in the proper mode.

1.1.2 Memory

MS-DOS controls main memory (RAM) as well as disk space and other devices. MS-DOS can load files into memory as data files or as executable files. The actual loading of files is performed by either IDOS.SYS or VDOS.SYS, the lowest level of the MS-DOS operating system. ICOMMAND.COM or VCOMMAND.COM supervises the loading of executable files depending in which mode you are.

Most programs return control to MS-DOS after normal or unexpected termination of the program.

MS-DOS can also overlay (write over) the transient portion of ICOMMAND.COM or VCOMMAND.COM to make room for a particularly large executable file. After such a file is executed, MS-DOS automatically loads the overlaid part of ICOMMAND.COM or VCOMMAND.COM back into memory from disk, and normal execution of ICOMMAND.COM or VCOMMAND.COM resumes. MS-DOS attempts to reload the transient portion from the root directory of the default drive at startup (not from the current default drive, if the default drive has changed) or from the path specified in COMSPEC in the ICONFIG.SYS/VCONFIG.SYS file.

If MS-DOS finds an incorrect version of ICOMMAND.COM or VCOMMAND.COM in attempting to load the overlaid section, it displays a message such as:

```
Invalid ICOMMAND.COM
  COMMAND.COM expected at X:\ICOMMAND.COM
  <CR> for retry, or new COMMAND.COM path =
```

where drive X is the original default drive, and VCOMMAND.COM can be substituted for ICOMMAND.COM depending on the current mode.

Insert the system diskette you used to load MS-DOS (or any diskette with the version of ICOMMAND.COM or VCOMMAND.COM that was originally loaded). To use another drive or path for reloading ICOMMAND.COM or VCOMMAND.COM, type the name of that drive or path name. Now MS-DOS will use that drive or path each time it reloads ICOMMAND.COM or VCOMMAND.COM during this operating session.

1.1.3 Disk Space

In MS-DOS, disk space is divided into four parts:

- ▶ The reserved sector contains boot information used each time MS-DOS is loaded.
- ▶ The directory contains information about each file on a given diskette. This information includes the file's complete filename, its size, and the time and date of its last modification.
- ▶ The file allocation table (FAT) contains location information for the data making up each file on a diskette.
- ▶ Files occupy the majority of disk space. An individual file does not necessarily reside in contiguous areas on the diskette; its contents might be "scattered" on the diskette so that disk space is not wasted.

1.2 System Configuration

With MS-DOS your system configuration can include the following peripheral devices:

- ▶ One parallel printer
- ▶ Two serial devices (a combination of MODEMs, printers, and so on)

To set up your operating system for your particular configuration, refer to the *PlusPC User's Guide*. Chapter 6 in this manual discusses the system configuration commands and the CAUX, CLST, and CTTY commands.

1.3 Loading MS-DOS

Before you start working with your system, you should have a backup copy of your MS-DOS system diskette. Refer to the *PlusPC User's Guide* for instructions on how to make backup copies of the diskette. To load MS-DOS, insert the system diskette in the drive as described in the *PlusPC User's Guide*.

Once MS-DOS is in main memory, it prompts you to enter the date:

```
Enter new date:_
```

MS-DOS keeps track of the date and time. If you enter the date and time when you load MS-DOS, it records the current date and time on each file that you create or change.

To set the date, follow these guidelines:

- ▶ Type the date using numbers separated by slashes or by hyphens, in one of these formats:

  —month/day/year or month-day-year (for American OS)

  —day-month-year or day/month/year (for European OS)

- ▶ For the month, enter a number from 1 to 12. For the day, enter a number from 1 to 31. For the year, enter a number from 1980 to 2099. Then press Return to submit your entry to MS-DOS.
- ▶ Use the Backspace key to correct mistakes.
- ▶ Do not type the name of the day because MS-DOS computes the day automatically.
- ▶ If you do not want to type in a date, press the Return key. Then MS-DOS uses the system creation date or the last valid date setting as the current date.

To enter the date of February 25, 1985, for example, you would type:

2/25/85 or 2-25-85 (American OS)

or

25-2-85 or 25/2/85 (European OS)

If the date you enter is not in the correct format or contains a number outside the valid ranges (such as 13 for the month), MS-DOS displays:

```
Invalid date:
Enter new date:_
```

After MS-DOS accepts your entry for the date, it displays its current time setting and prompts you to reset the time:

```
Enter new time:_
```

Note: The SHELL command in the CONFIG.SYS file can instruct MS-DOS to load the command processor without prompting you for the date and time at bootup.

To enter the time, use a 24-hour clock. For example, 1:00 pm is 13:00, and 11:00 pm is 23:00. When MS-DOS displays the time, it gives the hour, minute, second, tenths and hundredths of a second (0:00:00.00).

To change the time:

- ▶ Use the numbers 0–23 for the hour and 0–59 for the minutes. You do not have to make an entry for minutes or seconds. Do not type the tenths and hundredths of a second.
- ▶ Separate the time with colons (hour:minute:second) or with periods (hour.minute.second).
- ▶ Use the Backspace key to correct mistakes.
- ▶ Press the Return key when you want MS-DOS to accept your entry.

If you make an error, after you press the Return key MS-DOS displays:

```
Invalid time:
Enter new time:_
```

Retype the time.

After MS-DOS accepts your entry for the time, the MS-DOS command prompt and the cursor appear on your screen. By default, the prompt displays the name of the drive from which MS-DOS was loaded. For example, if MS-DOS is loaded from drive A, you will see one of the following prompts, depending on whether you are in I or V mode:

```
{ Mode I } A:\>_
```

or

```
Mode V A:\>_
```

Note that "Mode V" in the V mode prompt appears in reverse video.

Because the MS-DOS hierarchical directory structure functions the same for either mode, this manual refers to drive A as A>, drive B as B>, and so on.

The command prompt indicates that MS-DOS is in main memory and is ready for you to enter an MS-DOS command from the keyboard. You can change the MS-DOS prompt to be any letter(s), word, or phrase by using the PROMPT command (see Chapter 6.4 for more information on PROMPT).

You can check and change the date or time while you are using MS-DOS. Type the word **date** or **time** and press Return. MS-DOS shows the date or time and asks you to reset it. Press Return to leave it as is, or follow the rules for entering the date or time given in this section.

After MS-DOS accepts your entry for the time, the MS-DOS command prompt and the [illegible] appears. [illegible] the prompt shows the name of the drive from which MS-DOS was loaded. For example, if MS-DOS is loaded from drive A you will see one of the following prompts depending on whether you are in [illegible] mode:

```
  Mode 1 A>
```

or

```
Mode 2 A>
```

Note that "Mode 2" in the [illegible] appears in reverse video.

Because the MS-DOS [illegible] operate functions the same for either mode, [illegible] to drive A as A>, [illegible] and so on.

The command prompt indicates that MS-DOS is in main memory and is ready for you to enter an MS-DOS command from the keyboard. You can change the MS-DOS prompt to be any [illegible] word, or phrase by using the PROMPT command (see Chapter 6 for more information on PROMPT).

You can check and change the date and time while you are using MS-DOS. Type the word date or time [illegible] MS-DOS shows the date or time and [illegible]. Press Return to leave [illegible] or follow the [illegible] for entering the [illegible] given in this section.

The MS-DOS File System

This chapter describes some facts about MS-DOS files, including:

- ▶ Types of files you can use with MS-DOS
- ▶ Conventions for naming files
- ▶ Rules for using wild-card characters
- ▶ How to create and run batch files
- ▶ The function of the AUTOEXEC.BAT file
- ▶ How to use peripheral devices (device files) for input/output

MS-DOS stores and processes data as files. A file is made up of one or more related characters; the capacity of a file is limited only by the data storage capacity of a diskette or fixed (hard) disk volume. MS-DOS uses files as locations for writing or reading data. A file's location can be a disk, or a peripheral device, such as the keyboard, screen, or printer. The data in most files has been input from the keyboard.

2.1 Types of Files

MS-DOS recognizes these types of files:

- ▶ System files
- ▶ Command files (utility programs)
- ▶ Data/text files
- ▶ Batch files
- ▶ Source files

System files are predefined programs that:

▶ Run the computer and its devices

▶ Control other programs loaded into memory from disk, such as word processing programs

▶ Process data entered from the keyboard

A **command file** is a predefined program on disk. You run the program by typing the filename, followed by optional parameters. The terms "external command" and "utility program" both refer to command files. Utilities or external commands perform basic file and diskette management tasks (such as formatting and copying diskettes) that are necessary for every application. Each utility program is stored on disk with a filename extension of .COM or .EXE.

A **data/text file** is a collection of alphanumeric characters that can be in document form, such as a memo. It can also be a list of data, such as a mail list, payroll statement, or the output of an application program, like VBASICA.

A **batch file** is a sequence of MS-DOS commands. When you load a batch file, it performs a specific task or tasks (see Chapter 2.5).

A **source file** contains English-like program language statements that you create. Your computer uses a translation program called a compiler to put each source statement line into a format that the machine can process.

2.2 Filename Conventions

The filename conventions described in this section are rules for specifying the name of a file, the filename extension, and the drive name. If you follow clear and simple file-naming conventions, you will be able to easily identify the contents of your files.

2.2.1 Filenames

Names for MS-DOS files can be one to eight alphanumeric characters. The characters you can use are:

- ▶ The letters A through Z (either upper- or lowercase)
- ▶ The numbers 0 through 9
- ▶ The special characters ! @ # $ % & ( ) - ’ ˆ { } ˜ ‘

  Be careful when using some of these symbols in filenames, because you can reassign certain symbols to be switch characters. For more information on switch characters, see SWITCHAR in Chapter 6.3.1.

If you use any other characters, MS-DOS displays:

```
Invalid filename
```

Do not give any of your files the device names listed in Table 2-2.

2.2.2 Filename Extensions

You can add an optional filename extension to a filename. The extension is one to three alphanumeric characters preceded by a period. The valid characters for extensions are the same as for filenames.

You can use extensions to specify types of files. Table 2-1 lists some conventional MS-DOS file extensions and their meanings. Extensions other than those given in Table 2-1 or defined by your application program can be assigned at your discretion. For example, you might use your initials as the extension for the files you create, or the extension .TXT for text files or .MEM for memos.

If you specify more than three characters for the extension, MS-DOS accepts only the first three characters. If a data/text file has a filename extension, you must include the extension with the filename when you make a request for the file. To run a command or batch file, you omit the extension (.COM, .EXE, or .BAT) when you type the filename.

Table 2-1: MS-DOS File Extensions

FILE EXTENSION	MS-DOS INTERPRETATION
.ASM	8086 assembler language source code
.BAK	Backup file created by application
.BAS	BASIC source code (VBASICA)
.BAT	Batch command file
.COB	COBOL source code
.COM	Executable command file
.CRF	Cross-reference file
.DAT	Data file
.EXE	Relocatable executable file
.FOR	FORTRAN source code
.INT	Intermediate compiled code
.LIB	Library file
.LST	Listing of compilation or assembly
.MAP	Memory map from linker program
.OBJ	Relocatable object code module
.OVR	Overlay module
.PAS	Pascal source file
.PLM	PL/M source code
.PRN	Listing of compilation or assembly
.REF	Cross-reference listing
.$$$	Temporary system-generated file

2.2.3 Drive Names

The drive name is the letter name (A–O) of the disk drive where the file is currently located. Place a colon between the drive name and the filename itself. For example, to indicate that the file MAIL.LST is on drive B, type:

```
b:mail.lst
```

If you enter a filename without a drive name, MS-DOS searches for the file on the default drive.

To fully identify a file for MS-DOS, you might need to include the name of the subdirectory that contains the file. See Chapter 3 for information on using directory names.

The format for specifying a complete filename is:

```
drivename:filename.ext
```

The following description summarizes the characteristics of each part of the filename:

drivename:

- ▶ Is one letter (A–O)
- ▶ Is followed by a colon
- ▶ Does not need to be specified when the file is on the default drive

filename

- ▶ Contains up to 8 characters
- ▶ Does not use the symbols " / [ ] + = , ; : | < > \ (sp)

.ext

- ▶ Is preceded by a period (.)
- ▶ Contains up to 3 characters
- ▶ Does not use " / [ ] + = , ; : | < > \ (sp)
- ▶ Is optional except for some of the extensions in Table 2-1

2.3 Filename Wild-Card Characters

The ? and * wild-card characters replace existing characters in a filename. You can use wild-card characters to match a specified portion of a filename with existing filenames.

Wild-card characters can be used with these commands:

COPY	ERASE
DEL	PRINT
DIR	REN

The ? wild-card character used in a filename with an MS-DOS command matches any single existing character (or none) in that position.

For example, suppose your disk directory contains several similar filenames, such as:

```
CHAPT1
CHAPT2
CHAPT3
```

If you enter this command:

```
dir chapt?(cr)
```

MS-DOS displays only the filenames that match the first five characters of the filename in the command. The sixth character can be any character, as indicated by the ?.

The * wild-card character replaces any existing character in that position and all subsequent characters. A matching filename can be 2 to 8 characters long, and the other characters in the filename can be any character.

For example, if you have the sample directory shown above and you enter the following command, MS-DOS displays all the filenames that start with CH:

```
dir ch*(cr)
```

MS-DOS also displays any other filenames in your directory that begin with CH, such as CHARLES.MEM, CHART.DAT, or CHECKS.

You can also use the * to display all the files ending with a particular extension. For example, if you enter the following command:

```
dir *.bat(cr)
```

MS-DOS lists the names of all the batch files (indicated by .BAT) in the current directory.

You can specify *.* in a command to indicate all the filenames in a directory. For example, using *.* with the DEL command deletes all the files from the directory. **Note:** When you specify DEL *.*, MS-DOS asks if you are sure before it deletes the files. Make sure that you have backed up any files you need.

You can use ? and * together in a command. For example, if your directory contains:

```
83JOBS.PUB
82JOBS.PUB
83JOBS.ACT
```

You can enter this command:

```
dir 8?jobs.p*(cr)
```

MS-DOS displays these filenames:

```
83JOBS.PUB
82JOBS.PUB
```

83JOBS.ACT is not displayed, because .A does not match .P.

2.4 Device Names

MS-DOS can send data to or receive data from a device. MS-DOS interprets device names as the names of peripheral devices. You can use these device names as parameters in MS-DOS commands to refer to devices as if they were files. A device name, however, never has a file extension or drive name because device names are not disk files, but represent input/output devices. "Logical device" is another term for a device name.

The device names listed in Table 2-2 include the logical device names for the standard MS-DOS device drivers (programs that control the peripheral devices connected to your computer). Device drivers are contained on the system diskette. At bootup, drivers are loaded by the ICONFIG.SYS/VCONFIG.SYS file, and the drivers' names (logical device names) are displayed on the screen.

Table 2-2: MS-DOS Device Names

NAME	DEVICE TYPE
AUX COM1	Names for the device driver for serial port 1.
COM2	The name for the device driver for serial port 2.
CON	Keyboard input to the CPU, and CPU output to the screen.
LPT1 LST PRN	Names for the device driver for parallel port 1.
LPT2	The name for the device driver for parallel port 2 (reserved, but not in standard hardware).
LPT3	The name for the device driver for parallel port 3 (reserved, but not in standard hardware).
NUL	A nonexistent device for a command that requires a filename. Use this device when you do not want to create a file. You can also use it to get rid of unwanted console output.

For example, you can copy data from the CON device file (the keyboard) to a file on disk called PHN.LST:

```
copy con phn.lst(cr)
```

The next lines you type after the COPY CON command are temporarily stored in memory until you type CTRL-Z (ALT-Z for V mode) and press the Return key. Then the text you have entered is stored in the file PHN.LST.

You can also redirect output to a device by using the > symbol (described in Chapter 5). For example, this command sends the directory to the printer (PRN):

```
dir > prn(cr)
```

2.5 Batch Files

A batch file is a sequence of MS-DOS commands that perform a specific task. You can create batch files with the line editor EDLIN, with the COPY command (described in Chapter 6), or with any word processing program. You must give batch files the extension .BAT. To run the sequence of commands stored in a batch file, type the name of the file without the extension.

Using batch files, you can execute several MS-DOS commands by entering a single batch command. Because batch files can perform many different functions, writing batch files is similar to programming, and the files themselves are like short programs "written in MS-DOS." This chapter gives the basic information you need to start using and writing batch files; you will find examples of batch file usage throughout this manual.

The following example is a batch file named NEWDISK.BAT that contains a sequence of commands to display the directory of a formatted disk and check the diskette for inconsistencies. It also displays remarks (lines beginning with REM) that describe the batch file.

```
REM This is a file to check formatted disks
REM It is named NEWDISK.BAT
PAUSE Insert disk in drive B
DIR B:
CHKDSK B:
```

Once the NEWDISK.BAT file is saved on disk, you can run the two commands (DIR B: and CHKDSK B:) by typing the batch file name NEWDISK. PAUSE and REM are commands used only for batch files. These commands and the other commands used for batch files (FOR, IF, GOTO, ECHO, and SHIFT) are described in Chapter 6.4.

2.5.1 Batch File Conventions

Follow these rules when you use batch files:

1. To run a batch file, specify the batch file name (without the .BAT extension). If the batch file contains replaceable parameters, follow the filename with the values you want to substitute for the parameters (see Chapter 2.5.2).
2. If you press CTRL-C (ALT-C in V mode) while a batch file is being processed, MS-DOS displays:

   ```
   Terminate batch job (Y/N)?_
   ```

   Type Y if you want MS-DOS to ignore the rest of the commands in the batch file and return you to the system prompt. Type N to stop the current command being processed; MS-DOS then processes the next command in the batch file.
3. You can specify the name of another batch file as a command in the batch file. MS-DOS invokes that batch file from the one being processed.

2.5.2 Replaceable Parameters

Replaceable parameters are "dummy values" in a batch file. They are numbers preceded by a percent sign (such as %1 and %2). When you run a batch file, you include the value(s) you want to use in place of the replaceable parameters in the file.

You can specify ten replaceable parameters in a batch file (%0 to %9). %0 is always replaced by the drive name (if one is designated) and the name of the batch file. %1 is replaced by the first value following the filename in the command you enter to run the batch file. %2 is replaced by the second value, and so on. You can specify more than ten replaceable parameters by using the SHIFT command.

For example, you can use the COPY CON command to create a phone list file:

```
copy con phn.lst(cr)
Mary Smith (415) 123-4567(cr)
Engineering (408) 111-1111(cr)
George Jones (916) 555-5555(cr)
Marketing (408) 777-7000(cr)
^Z(cr)
```

Then you can create a batch file named LOOKING that uses the FIND command to look through PHN.LST for a dummy value (the replaceable parameter %1):

```
copy con looking.bat(cr)
find "%1" phn.lst(cr)
^Z(cr)
```

When you load the batch file, include the value or name that you want to substitute for the replaceable parameter in the batch file. For example, if you specify:

```
looking 408(cr)
```

MS-DOS replaces the parameter %1 with 408. MS-DOS then processes the LOOKING batch file using this value and displays the command from the file:

```
A>find "408" phn.lst
```

The FIND command then displays all the lines from the PHN.LST file that contain 408:

```
---------- phn.lst
Engineering (408) 111-1111
Marketing (408) 777-7000
```

2.5.3 The AUTOEXEC.BAT File

AUTOEXEC.BAT is the name of a batch file that MS-DOS always looks for and processes immediately after it loads into memory. If MS-DOS does not find an AUTOEXEC.BAT file, MS-DOS prompts you for the date and time and then displays its command prompt. When there is an AUTOEXEC.BAT file, MS-DOS displays the date and time prompts only if the DATE and TIME commands (described in Chapter 6.4) are included in the AUTOEXEC.BAT file.

Create an AUTOEXEC.BAT file if you want a command or sequence of commands to operate each time MS-DOS loads. For example, suppose you are in V mode and your principal application is VBASICA. You can use COPY CON to create an AUTOEXEC.BAT file that takes you straight into VBASICA when you boot up:

```
copy con autoexec.bat(cr)
basica(cr)
^Z(cr)
```

The command "BASICA" tells MS-DOS to load the VBASICA language into memory. If this AUTOEXEC.BAT file and the command file are on your system diskette, then each time MS-DOS loads into memory it displays the first command from the file:

```
A>basica
```

VBASICA loads into memory and displays its command prompt (Ok). You can now use VBASICA. To return to MS-DOS, use the VBASICA command "SYSTEM" to exit to the operating system.

If you create an AUTOEXEC.BAT file and want to process it immediately, enter this command:

autoexec(cr)

or reload MS-DOS either by pressing the reset button or by pressing the CTRL, ALT, and DEL keys at the same time.

The MS-DOS Subdirectory System

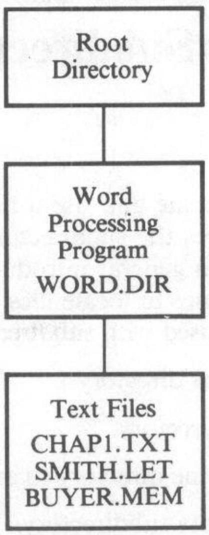
MS-DOS allows you to create and use a hierarchical subdirectory system. This chapter describes the subdirectory system and how to work with it. The first topic is a general introduction, followed by the rules for using subdirectory names to locate files. Then this chapter describes the MS-DOS commands used with subdirectories:

- ▶ DIR displays the current directory.
- ▶ MKDIR creates a subdirectory.
- ▶ CHDIR changes from one directory to another.
- ▶ RMDIR deletes an empty subdirectory.

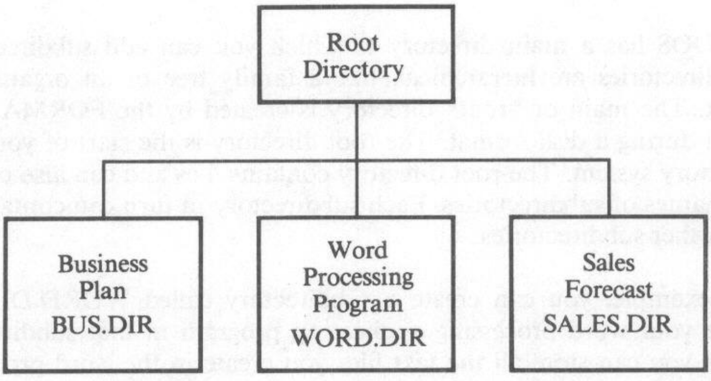
A **directory** is a list of the files on a diskette, including the filenames, the file size, and the date and time each file was created. MS-DOS enters the filename and date and time in the directory each time you create or update a file.

MS-DOS has a main directory to which you can add subdirectories. Subdirectories are hierarchical, like a family tree or an organization chart. The main or "root" directory is created by the FORMAT program during a disk format. The root directory is the start of your subdirectory system. The root directory contains files and can also contain the names of subdirectories. Each subdirectory in turn can contain files and other subdirectories.

For example, you can create a subdirectory called WORD.DIR and place your word processing application program in that subdirectory. Then you can store all the text files you create in the word processing subdirectory, WORD.DIR, as shown in the following diagram.



Similarly, you can create a different subdirectory for each application program, and group your files in the appropriate subdirectory:



When you save a file, MS-DOS puts the filename in the directory you are currently using or in another directory that you specify. A file in one directory can have the same name as a file in another directory because MS-DOS keeps each directory separate.

You can search for a file in one subdirectory rather than searching through many kinds of files in one large main directory. Chapter 3.1 describes how to specify the subdirectory name (or path) for a file.

3.1 Using Directory Names (Paths) to Locate Files

When you use hierarchical directories, you must tell MS-DOS where to find your files in the directory structure. In addition to the filename, you give the name of the subdirectory where the file is located.

Giving the subdirectory name means specifying a **path** for MS-DOS to follow through the directory hierarchy. A path or subdirectory name:

- ▶ Is a sequence of directory names, each separated from the previous one by the path-separator character, usually a backslash (\). (The path-separator character becomes a slash (/) if you change the switch character to any character other than a slash. For more information, see the SWITCHAR command in Chapter 6.3.1.)
- ▶ Leads MS-DOS through the directory hierarchy. Each \ leads to a lower subdirectory level.
- ▶ Can end with either a filename or a directory name.
- ▶ Can go through only existing subdirectories.
- ▶ Must not exceed 63 characters.

The format for a path to a file or subdirectory is:

```
[drivename:][[\]directoryname[\directoryname...]\][filename.ext]
```

drivename
: is the drive containing the file or directory you want. You can omit the drive name if the file or directory is on the default drive.

\
: The initial backslash always indicates the root directory. If a path begins with a backslash, MS-DOS starts searching for the file or directory at the root directory. Otherwise, MS-DOS searches downward from the current directory.

directoryname
: is the name of a subdirectory (created with the MKDIR command, described in Chapter 3.3). If the first subdirectory name is not preceded by a backslash, it is directly below the current directory. Each additional \directoryname indicates another subdirectory level below the root directory or below the current directory.

filename.ext
: is the full filename and extension of the file you want. The filename is always last in the path. If the file is in the current directory, you do not need to specify a path with the filename.

MS-DOS recognizes three abbreviated pathnames:

.
: is shorthand for the current directory. You can use the . notation with the DIR command to display the current subdirectory, although it is not necessary.

..
: is shorthand for the current directory's parent directory. MS-DOS uses the . and .. directories to create the hierarchical tree directory structure. You can use the .. notation with DIR, CHDIR, and DEL.

\
: is shorthand for the root directory. You can use the \ notation with the CHDIR command.

3.1.1 Examples of Paths

The following examples demonstrate different kinds of pathnames.

```
\WORD.DIR
\MGR1.NAM
```

These are subdirectory names. The initial \ indicates the root directory. Both WORD.DIR and MGR1.NAM are first-level subdirectories (directly below the root directory).

```
\WORD.DIR\CHAP1.TXT
\MGR1.NAM\JOBS\CLERK
```

These are full pathnames. In the first example, the file or subdirectory CHAP1.TXT is in the first-level subdirectory WORD.DIR. In the second example, the file or subdirectory CLERK is in the subdirectory JOBS under the first-level subdirectory MGR1.NAM.

```
WORD.DIR\DAVID
JOBS\BENEFITS
```

These are relative pathnames that refer to the current directory level. In the first example, DAVID is a file or directory in the subdirectory WORD.DIR below the current directory. In the second example, the file or directory BENEFITS is in the JOBS subdirectory, which is listed in the current directory. If the current directory is the root directory, these pathnames are the same as specifying \WORD.DIR\DAVID and \JOBS\BENEFITS.

3.1.2 Working with Files in Subdirectories

You can use paths or subdirectory names with any of the MS-DOS commands that operate on filenames. This section gives several examples of using pathnames in MS-DOS commands.

You locate a file by specifying a path for MS-DOS to follow through the directory hierarchy. The filename is always the last name in the path. When the file is in the current directory, you can omit the path and give just the filename. To get to a file in the previous directory, you can use .. as the path.

Displaying Files Onscreen

You can display the contents of text files on your screen using the TYPE command. For example, to display the CHAP1.TXT file in the WORD.DIR subdirectory on the default drive, enter:

```
type \word.dir\chap1.txt(cr)
```

In the preceding example, the first \ tells MS-DOS to start the search for the file at the root directory. MS-DOS finds the subdirectory WORD.DIR listed in the root directory. In the WORD.DIR subdirectory, MS-DOS finds the CHAP1.TXT file and displays the file on the screen.

If WORD.DIR is the current directory, you can specify the filename without a path, like this:

```
type chap1.txt(cr)
```

If you include the drive name in a command, the drive name must precede the path. For example, this command displays CHAP1.TXT from the WORD.DIR subdirectory on drive B:

```
type b:\word.dir\chap1.txt(cr)
```

Copying Files

You use the internal command COPY to copy files. To copy a file from one directory to another, you must specify the path to the file to be copied and the path to the directory where the file is to be copied. For example, to copy the CHAP1.TXT file from the WORD.DIR subdirectory on drive A to the root directory on drive B, and to rename the file INTRO, enter:

```
copy a:\word.dir\chap1.txt b:\intro(cr)
```

Deleting Files

To delete files from a directory, you use the internal command DEL (Delete). If the file you want to delete is in the current directory, you do not need to specify a path to the file. For example, if you are using the WORD.DIR directory, you can delete a file from that directory with the command:

```
del chap1.txt(cr)
```

If you are using any other directory, you must include the path to the file you want to delete from WORD.DIR. For example, this command deletes CHAP1.TXT from the WORD.DIR subdirectory on drive B:

```
del b:\word.dir\chap1.txt(cr)
```

You can use the .. notation with DEL to refer to the previous directory. For example, this command deletes MYFILE from the parent directory of the current directory:

```
del ..\myfile(cr)
```

3.1.3 Paths and External Commands

When you enter a command at the system prompt, the PATH command searches your directories for the program file. PATH searches only for .EXE, .COM, and .BAT files. Text files and other types of files cannot be accessed using the PATH command.

You do not need to move to the subdirectory where the program files are contained. Instead, a path has been provided on your system diskette to the subdirectories where the correct program files are located. See Chapter 1.1.1 for an explanation of how the PlusPC MS-DOS system diskette stores program files.

Whenever you enter an external command, MS-DOS searches the current directory and then the subdirectory that was specified with the PATH command. If you specify PATH while in one mode, you still need to specify another PATH when you enter the other mode. If you reboot to the same mode, or switch to the other mode, you will have to reset PATH (see the PATH command in Chapter 6.4).

3.2 Listing a Directory

The DIR command, described in Chapter 6.4, lists the contents of a directory. You can list a directory onscreen or send it to a device or file. To list a subdirectory with DIR, you can use the path notation shown in Chapter 3.1.1. For example, you can display the current directory on the default drive:

```
dir(cr)
```

To display the parent directory (one level above the current directory), you can use the shorthand notation:

```
dir ..(cr)
```

If you enter the DIR command from the WORD.DIR subdirectory, MS-DOS displays that subdirectory. To display the WORD.DIR subdirectory on the default drive (regardless of the directory you are in), enter:

dir \word.dir(cr)

The display for the WORD.DIR subdirectory might look like this:

```
Volume in drive A has no ID
Diskette is V format
Directory of A:\word.dir

.            <DIR>          5-05-85  10:09a
..           <DIR>          5-05-85   2:15p
DAVID.L      <DIR>          5-20-85   3:40p
CHAP1.TXT             3345  6-15-85  11:13a
        4 File(s)   370688 bytes free
```

This DIR display gives the following information:

▶ No Volume ID was assigned to the diskette during formatting or with the VOL command.

▶ The diskette is V (VICTOR) format.

▶ This directory is WORD.DIR, a first-level subdirectory.

▶ The single period (.) indicates the current directory. The . is displayed only for subdirectories, not for the root directory. MS-DOS automatically creates the . entry when you create a directory.

▶ Two periods (..) indicates the parent directory (the subdirectory level above this one).

▶ The WORD.DIR subdirectory has another subdirectory (DAVID.L) below it.

▶ WORD.DIR contains the file CHAP1.TXT, which is 3345 bytes, and was created or last modified at 11:13 am on June 15, 1985.

3.3 Making a Subdirectory

You can create a subdirectory by entering a MKDIR (Make Directory) command from the keyboard or from a batch file. See the descriptions of the MKDIR and FOR commands in Chapter 6.4. You can shorten MKDIR to MD. In the MKDIR command, give the name of the directory you want to create. MS-DOS creates a subdirectory as a special kind of file with a < DIR > attribute.

The conventions for naming a directory are the same as for naming a file. You can use 1 to 8 alphanumeric characters, and you can include an optional filename extension of 1 to 3 alphanumeric characters preceded by a period.

For example, suppose you want to create a subdirectory named MGR1.NAM under the root directory on the diskette in the default drive. If you are in the root directory, you can enter:

```
mkdir mgr1.nam(cr)
```

If you are in another subdirectory, you can enter:

```
md \mgr1.nam(cr)
```

You can verify that MS-DOS created the subdirectory by entering the DIR command.

Once a directory exists, you can add other subdirectories to it. For example, you can create a subdirectory called JOBS beneath MGR1.NAM. Enter:

```
md \mgr1.nam\jobs(cr)
```

You can also create a subdirectory by first moving into the parent directory, and then making the subdirectory. You move from one directory to another with the CHDIR command, described in Chapter 3.4. For example, enter these commands:

```
chdir mgr1.nam(cr)
mkdir jobs(cr)
dir(cr)
```

MS-DOS moves to the MGR1.NAM subdirectory, creates the JOBS directory, and displays the current directory, MGR1.NAM:

```
Volume in drive A has no ID
Diskette is V format
Directory of A:\mgr1.nam

.            <DIR>      6-30-85      9:00a
..           <DIR>      8-09-85     10:09a
JOBS         <DIR>      8-09-85     10:34a
        3 File(s)    370688 bytes free
```

If you create and save a file now, MS-DOS places it in the MGR1.NAM subdirectory. For example, if you enter this COPY CON command, MS-DOS creates and saves a file named TITLES in the MGR1.NAM subdirectory:

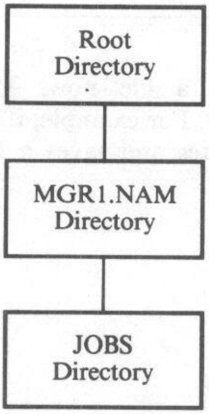
```
copy con titles(cr)
Publication Manager(cr)
Managing Editor(cr)
Copy Editor(cr)
^Z(cr)
```

Now the name of the TITLES file, its size, and the date and time it was created are displayed in the MGR1.NAM directory:

```
 Volume in drive A has no ID
 Diskette is V format
 Directory of A:\mgr1.nam

.             <DIR>        6-30-85     9:00a
..            <DIR>        8-09-85    10:09a
JOBS          <DIR>        8-09-85    10:34a
TITLES                51   8-09-85    11:20a
        4 File(s)   370637 bytes free
```

The full path to JOBS is \MGR1.NAM\JOBS, and the path to the TITLES file is \MGR1.NAM\TITLES. Each \ indicates a level below the root directory. MGR1.NAM is the first subdirectory (indicated in the directory by the .), and JOBS is the subdirectory beneath it:

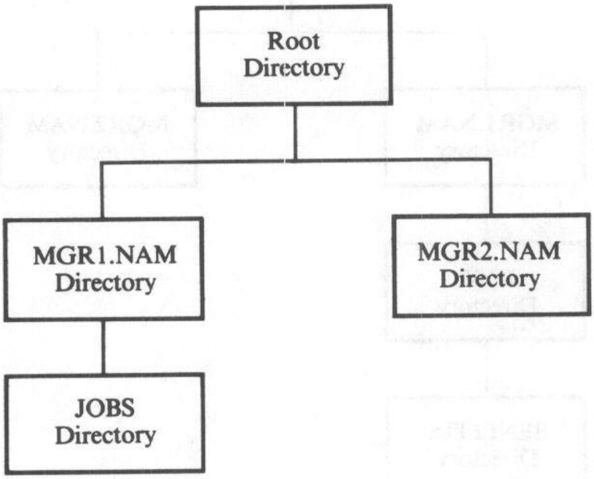


To add another subdirectory, such as MGR2.NAM, at the same level as MGR1.NAM, you must tell MS-DOS to start at the root directory:

mkdir \mgr2.nam(cr)

The first \ in a MKDIR command indicates the root directory to MS-DOS. You can omit the initial \ if you are at the root directory.

MS-DOS creates MGR2.NAM and puts the new directory name in the root directory:



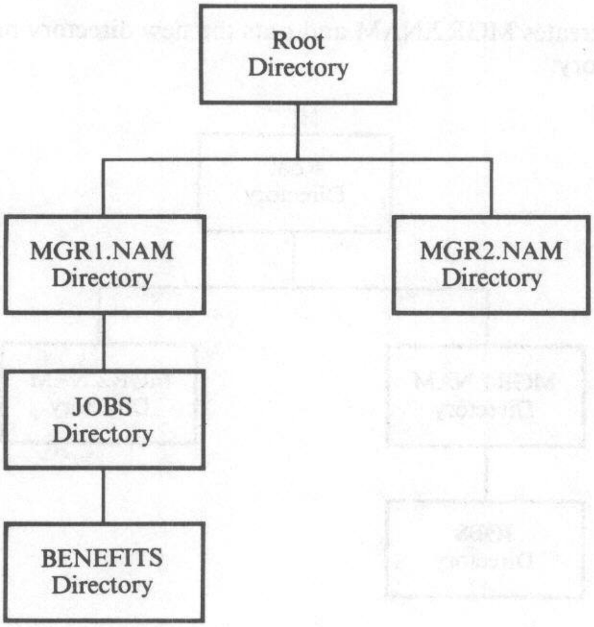
You can add a third directory level beneath JOBS, such as a subdirectory named BENEFITS. If you are currently at the JOBS subdirectory, you can create BENEFITS by entering:

```
mkdir benefits(cr)
```

If you are not at JOBS, enter:

```
mkdir \mgr1.nam\jobs\benefits(cr)
```

Now you have the following subdirectory system:



3

Because each subdirectory name is listed in the preceding directory, you cannot create a new subdirectory unless the parent directory exists. If you load MS-DOS and try to create several new subdirectories in the same command, such as:

mkdir mgr2.nam\mgr3.nam\mgr4.nam(cr)

MS-DOS responds with:

```
Unable to create directory
A>_
```

Likewise, MS-DOS does not allow you to give a file the same name as its subdirectory. This is because subdirectories are listed as files in a directory, and duplicate filenames are not allowed.

3.4 Moving into a Directory

Using a hierarchical file structure gives you two means of locating files. You can enter a command in any directory and locate the file by using a pathname, or you can move to the directory in which the file is located and enter the command there.

You move into a directory to:

- ▶ Display the directory
- ▶ Create files or subdirectories
- ▶ Delete files
- ▶ Use files or programs in the directory
- ▶ Remove subdirectories

The CHDIR (Change Directory) command tells MS-DOS to move from one directory to another. You can shorten CHDIR to CD.

If you enter the CHDIR command without any parameters, MS-DOS displays the name of the current directory. Entering CHDIR with a subdirectory name moves you to that directory if it is listed in the current directory. If you enter CHDIR and a subdirectory name that is not listed in the current directory, MS-DOS displays:

```
Invalid directory
```

Using the examples in Chapter 3.3, if you are in the root directory, you can move into the next subdirectory (MGR1.NAM) by entering one of the following commands:

```
chdir mgr1.nam(cr)
cd \mgr1.nam(cr)
```

The first \ in a CHDIR command indicates the root directory. If a directory name follows the backslash, it names the first subdirectory below the root directory. Each additional \ in the command is another subdirectory level in the path to the specified directory. The command:

```
chdir \mgr1.nam\jobs\benefits(cr)
```

places you in the BENEFITS subdirectory. To move up to the MGR1.NAM subdirectory, enter:

```
chdir \mgr1.nam(cr)
```

To get to JOBS from MGR1.NAM (now the current directory), enter:

```
chdir jobs(cr)
```

If you are not at MGR1.NAM, you must indicate the path to JOBS:

```
chdir \mgr1.nam\jobs(cr)
```

You can move to the level above the current directory level by specifying .. with CHDIR:

```
chdir ..(cr)
```

This command moves you to the directory above the one you are now using. For example, if you are currently at JOBS, MS-DOS moves up to MGR1.NAM.

Entering CHDIR followed by a \ always returns you to the root directory. You should enter this command when you finish working in a subdirectory; if you do not, you might save files in the wrong subdirectory. CHDIR \ ensures that MS-DOS is at the root directory.

3.5 Removing a Subdirectory

The RMDIR (Remove Directory) command removes an empty subdirectory from the hierarchy. You can abbreviate RMDIR to RD.

Before you can remove a subdirectory, you must delete all filenames (with DEL) and any lower subdirectories (with RMDIR). An empty directory lists only the . and .. notations, which cannot be deleted because they are part of the MS-DOS hierarchy system.

You can remove only one subdirectory at a time. For example, if you want to remove the BENEFITS subdirectory, first delete any files in BENEFITS. If you are in the BENEFITS subdirectory, use CHDIR to move to another directory. To remove the BENEFITS subdirectory if you are in the JOBS subdirectory, enter:

```
rmdir benefits(cr)
```

You can also specify a full path from the root directory. For example, this command removes the BENEFITS subdirectory from the JOBS subdirectory if BENEFITS is already empty:

```
rmdir \mgr1.nam\jobs\benefits(cr)
```

If you are currently using the MGR1.NAM directory, you can specify the path to the empty BENEFITS subdirectory and remove it:

```
rmdir jobs\benefits(cr)
```

If the directory you name does not exist, or if you try to remove the current directory or one that is not empty, MS-DOS displays:

```
Invalid path, not directory,
or directory not empty
```

Using the Line Editor (EDLIN)

EDLIN is a line editor program used mainly to create and edit source files (such as MS-DOS batch files). You can also use EDLIN to create or edit text. To input text such as memos or documents, however, it is simpler and more efficient to use a word processing program.

You can use two types of commands for editing with EDLIN:

▶ File commands, which create or edit files

▶ Line commands, which edit characters within a single line of a file

The file commands are unique to EDLIN. The line commands are the MS-DOS command-line editing functions (described in Chapter 5); these differ depending on the mode you are using.

To use EDLIN, load the program as shown in Chapter 4.1. Chapter 4.1.1 shows you how to create a batch file; this batch file is used in examples throughout the chapter. Chapter 4.1.2 shows you how to edit an existing file. Chapter 4.2 discusses how to use the EDLIN file commands and introduces the format of the commands. Chapter 4.2.3 summarizes the EDLIN file commands.

4.1 Invoking EDLIN

The EDLIN.COM file must be available on disk in order to load the program. If you are using subdirectories, MS-DOS must be able to find EDLIN.COM in the directory system. If it is not in the current subdirectory, give its subdirectory location in the EDLIN command, or enter a PATH command to indicate its location.

You load EDLIN and the file you want to edit at the same time:

edlin filename(cr)

The filename can be either the name of an existing file or the name of the file you are creating. Include the drive name if the file is not on the default drive. To edit a data/text file in a subdirectory, either change to that subdirectory before loading EDLIN or give the subdirectory name with the filename when you invoke EDLIN. Chapter 3 describes how to use the MS-DOS subdirectory system.

4.1.1 Creating a File

4

To create a new file, enter EDLIN followed by the name of a file that does not exist. Include the drive on which the file is to be saved unless you want to save it on the default drive.

For example, to create the file NEWDISK.BAT in subdirectory MGR1.NAM on the diskette in drive B, enter:

edlin b:\mgr1.nam\newdisk.bat(cr)

EDLIN creates a new, empty file and displays the message "New file" and the command prompt (*):

```
New file
*_
```

To enter text into the file, type the Insert command (I) at the * prompt and press Return. EDLIN then numbers and displays the first line for your entry:

```
New file
*I(cr)
          1:*_
```

Now you can type in the text of the file. End each line with a Return. EDLIN then displays the next line number.

Here is an example of an MS-DOS batch file created with EDLIN. This file displays the directory of a formatted disk and checks the disk directories for consistency.

```
A>edlin b:\mgrl.nam\newdisk.bat(cr)
New file
*I(cr)
    1:REM This is a file to check formatted
      disks(cr)
    2:REM It is named NEWDISK.BAT(cr)
    3:PAUSE Insert formatted disk in drive B(cr)
    4:DIR B:(cr)
    5:CHKDSK B:(cr)
```

You can stop text entry without saving any of the file by entering the Quit (Q) command. When you finish entering data, type a CTRL-C (Interrupt) or CTRL-Z (Terminate Text Entry command) on the next line and press Return. Either CTRL-C or CTRL-Z stops Insert mode and returns you to the * prompt. (If you are using EDLIN in V mode, substitute ALT-C for CTRL-C, and ALT-Z for CTRL-Z.)

To display the file you created, enter the List command (L). EDLIN displays the file and numbers the lines. (Line numbers are not saved in the file on disk.) When you insert new lines or delete existing lines, EDLIN adjusts the line numbers accordingly. The line numbers always run from 1 through the number of the last line.

To save the file on disk, type the End command (E). You are returned to the MS-DOS command prompt.

4.1.2 Editing an Existing File

To edit an existing file, specify EDLIN, the drive and subdirectory containing the file, and the filename with its extension. For example, if you want to make changes to the file NOTICE.MEM in the WORD.DIR subdirectory on the default drive, enter:

edlin \word.dir\notice.mem(cr)

EDLIN loads the file into memory. If EDLIN can load the entire file, it displays the following message and the asterisk command prompt (*):

```
End of input file
*_
```

If EDLIN cannot load the entire file, it fills up 3/4 of available memory with the first part of the file and does not display the "End of input file" message.

To edit a file larger than memory:

1. Display the portion of the file in memory with the List Line command (L, described in Chapter 4.2.3).
2. Edit the portion of the file in memory.
3. Save the edited portion of the file on disk with the Write command (W, described in Chapter 4.2.3). Add the rest of the lines from the file by using the Append command (A, described in Chapter 4.2.3).

4.2 Using File Commands

Use the Insert command (I) to create an EDLIN file, as described in Chapter 4.1.1. After the file is created, you can use the EDLIN file commands to:

▶ Add lines from disk to memory if the file is too large for memory (Append).

▶ Delete one or more lines from the file (Delete).

▶ Modify existing lines (Edit).

▶ Add one or more lines of text to a file (Insert).

▶ Display the series of lines that you want to change (List).

▶ Replace previous text with new text within a certain number of lines in the file (Replace).

▶ Search through a specified number of lines in a file for a string of characters that you want to change (Search).

▶ Write edited portions of a large file from memory to disk (Write).

File commands are single letters (such as A for Append) with optional parameters. The parameters select, display, or change one or more lines in a file. You type the letter of the command and the parameters you want from the keyboard. The parameters and the commands are summarized in Table 4-1 and in Chapter 4.2.3.

4.2.1 File Command Parameters

The file command parameters are shown in Table 4-1.

Table 4-1: EDLIN File Commands and Parameters

COMMAND PARAMETER	PARAMETER DESCRIPTION	COMMANDS THAT USE PARAMETER
n	Number of lines.	
line	Represents the line number you select. For the line number, you can specify a decimal integer from 1–65534. If the number is larger than the lines in memory, the number indicates the line after the last existing line. Line numbers in a command must be separated from each other by a comma or a space. Lines can also be specified by:	A—Append lines W—Write lines
	. (a period) represents the current line (marked on the screen by an * between the line number and the first character of the line).	D—Delete lines N—Edit lines I—Insert lines L—List text R—Replace a string of characters
	# (the pound sign) specifies the line after the last line (same as specifying the number larger than the last line number in the file).	
	(cr) (Return) directs EDLIN to use a default line number appropriate to the command.	
?	Directs EDLIN to query for a yes or no response to an Ok? prompt.	R—Replace a string of characters S—Search for a string
string	Represents one or more characters to be found or replaced. Each string must end with a CTRL-Z or a Return. There should be no spaces between the string parameters unless the space is part of the string. Do not enter spaces between string parameters and the command.	R—Replace a string of characters S—Search for a string in a line

4.2.2 File Command Conventions

Some conventions are common to all the file commands. With any file command you can:

▶ Type commands with or without a space between the line number and the command. For example:

```
5I
```

can also be typed:

```
5 I
```

▶ Reference lines relative to the current line.

On the screen, the current line has an * between the line number and the first character of the line. You include a plus sign with a line number in a command to indicate the number of lines following the current line. A minus sign with a line number in a command indicates the number of lines before the current line.

For example, this command directs EDLIN to begin inserting text one line before the current line on the screen:

```
- 1I
```

▶ Issue multiple commands on one command line. Separate multiple commands in a line with a semicolon. For example, if you want to edit a line and then display other lines, enter:

```
10; - 2, + 1L(cr)
```

This command lets you edit line 10, then display (List) the two lines before line 10 and the line following line 10. The comma after − 2 is a syntax element for "line,line"—see the List command in Chapter 4.2.3.

If you are issuing a multiple command with a string parameter, as in a Search or Replace command, separate the commands with a CTRL-Z.

For example, this command:

```
2,10 SThis string^Z – 2; + 2L(cr)
```

searches for the first occurrence of the characters "This string" in lines 2 through 12. It then displays the 2 lines before the line that contains "This string" and the 2 lines after it.

If "This string" cannot be found, the 2 lines before the current line (noted on the screen with the *) and the 2 lines after the current line are displayed.

4.2.3 Descriptions of File Commands

4

This section presents detailed descriptions of each EDLIN command for editing files. The descriptions are given alphabetically.

Append Lines (A)

```
[n]A(cr)
```

A adds the number of lines, specified by n, from disk to the end of the lines currently in memory.

n represents the number of lines.

Use this command for large files that cannot fit into memory at one time. Edit the lines in memory, write them to disk with the Write command (W), then use the Append command (A) to put the rest of the file in memory.

If you enter a number with the A command, that number of lines is appended to the part of the file currently in memory. If no number is entered, the lines are read into memory until memory is 3/4 full. If memory is already full, no more lines can be appended.

When the last line of the file is in memory, EDLIN displays:

```
End of Input File
*_
```

Delete Lines (D)

[line][,line]D(cr)

D deletes the lines specified. "Line" can be any of the parameters for a line described in Table 4-1. Without parameters, D deletes the current line.

line
: deletes the specified line.

,line or ,#
: deletes the current line (marked by the *), the specified line, and all lines between the current line and the specified line.

line, or line,line or line,#
: deletes the specified lines and all lines between them.

The line immediately after the deleted lines becomes the current line (marked by the *).

For example, to delete lines 2 and 3 in the NEWDISK.BAT file:

```
1:REM This is a file to check formatted disks
2:REM This is a batch file created by EDLIN
3:REM This file is named NEWDISK.BAT
4:PAUSE Insert formatted disk in drive B
5:DIR B:
6:CHKDSK B:
```

Enter:

2,3 D(cr)

To verify the result, specify the List command (L):

```
*L(cr)
1:REM This is a file to check formatted disks
2:*PAUSE Insert formatted disk in drive B
3:DIR B:
4:CHKDSK B:
```

After lines 2 and 3 are deleted, line 4 becomes line 2 and is marked by the asterisk as the current line.

To delete a single line, such as line 3, enter:

3D(cr)

To see the result, specify the List command (L):

```
*L(cr)
1:REM This is a file to check formatted disks
2:PAUSE Insert formatted disk in drive B
3:*CHKDSK B:
```

When line 3 is deleted, line 4 becomes line 3 and is marked as the current line.

To delete the current line, now marked as line 3, enter:

D(cr)

Specify the L command at the * prompt, and the result is:

```
  1:REM This is a file to check formatted disks
  2:PAUSE Insert formatted disk in drive B
```

Edit Line

[n](cr)
[.](cr)
[#](cr)

n represents the line number.

. indicates the current line.

indicates the line after the last line in the file.

See Table 4-1 for an explanation of the line number parameters.

EDLIN displays the line number and the text on that line. On the next line, it repeats the line number and displays the asterisk prompt and the cursor. You can start editing the line at the cursor position. You can use any of the command-line editing keys described in Chapter 5 to edit the line. Note that the command-line editing keys are different in V mode and I mode.

If you do not want to change the current line (marked by the *), and the cursor is at the beginning of the line, press the Return key to accept the line as it is. If you press Return while the cursor is in the middle of the line, the rest of the line is deleted.

For example, you can select line 1 with the Edit Line command:

```
*1(cr)
```

EDLIN displays the specified line:

```
1:*REM This is a file to check formatted disks
1:*_
```

You can now type in a new line 1, or you can use the line commands to edit the current line (the template, described in Chapter 5).

End Edit (E)

E(cr)

E ends the editing session, saves the edited file on disk, changes the original file's extension to .BAK, and exits to MS-DOS. If the file was newly created during the editing session, no .BAK file is created.

Because the E command takes no parameters, you cannot select the drive on which to save the file. If you did not designate the drive when you invoked EDLIN, such as EDLIN B:NEWDISK, the file is saved on the default drive.

If the disk does not have enough free space for the entire file to be written, the write is aborted and the edited file is lost. Part of the file, however, may be written to the disk.

Insert Lines of Text (I)

```
[line]I(cr)
```

I inserts line(s) of text immediately before the specified line.

"Line" can be any of the parameters for a line described in Table 4-1.

You must enter the I command to create a new file. The first insert begins with line number 1. EDLIN numbers each new line sequentially each time you press Return. EDLIN remains in insert mode until you enter a CTRL-Z or a CTRL-C as your last line (ended with a carriage return).

If you insert lines between existing lines in a file, all line numbers following the inserted section are incremented by the number of lines inserted.

If you do not include a line number, the new lines are inserted immediately before the current line. If the specified line number is larger than the last line number or if # is specified as the line number, the inserted lines are appended to the end of the file. In this case, the last line inserted becomes the current line.

For example, to add lines before line 2 in the following file:

```
1:REM This is a file to check formatted disks
2:PAUSE Insert formatted disk in drive B
3:DIR B:
4:CHKDSK B:
```

Enter:

```
2 I(cr)
```

EDLIN displays:

```
2:*_
```

Enter:

```
2:REM This file is a batch file(cr)
3:REM This batch file is called NEWDISK(cr)
4:^Z(cr)
```

You can display the new file with the L command:

```
1:REM This is a file to check formatted disks
2:REM This file is a batch file
3:REM This batch file is called NEWDISK
4:*PAUSE Insert formatted disk in drive B
5:DIR B:
6:CHKDSK B:
```

To insert lines immediately before the current line (line 4 in the previous example), enter:

I(cr)

The screen displays:

```
4:*_
```

If you now enter:

```
4:*REM NEWDISK was created with EDLIN(cr)
5:*^Z(cr)
```

and type the L command to list the file, the result is:

```
1:REM This is a file to check formatted disks
2:REM This file is a batch file
3:REM This batch file is called NEWDISK
4:REM NEWDISK was created with EDLIN
5:*PAUSE Insert formatted disk in drive B
6:DIR B:
7:CHKDSK B:
```

List Lines (L)

[line][,line]L(cr)

"Line" can be any of the parameters for a line described in Table 4-1.

L lists the specified range of lines, as follows:

- ▶ With no parameters, L lists the 23 lines centered around the current line (marked by an *). The 23 lines are the 11 lines preceding the current line, the current line, and the 11 lines following the current line.
- ▶ line,line or line,# lists the specified range of lines.

- ,line lists the range of lines preceding the specified line.
- line, lists a range of lines following the specified line.

Assume the following file exists and is ready to edit:

```
1:REM This is a file to check formatted disks
2:REM It is named NEWDISK.BAT
3:PAUSE Insert formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

To list the entire file, enter:

L(cr)

All five lines are displayed, because L without parameters displays up to 23 lines of text.

To list a range of lines preceding line 3, enter:

,3L(cr)

The result is:

```
1:REM This is a file to check formatted disks
2:REM It is named NEWDISK.BAT
3:PAUSE Insert formatted disk in drive B
```

To list a range of lines following line 3, enter:

3,L(cr)

The result is:

```
3:PAUSE Insert formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

Quit Edit (Q)

Q(cr)

Q quits the editing session without saving the editing changes, and exits to MS-DOS. The Q command is a quick way to exit an editing session. When the Q command is given, EDLIN displays the message:

```
Abort edit (Y/N)?
```

Press Y to quit the editing session; press N (or any other key except CTRL-C) to continue the editing session.

Replace String (R)

[line][,line][?]Rstring1[^Z]string2(cr)

R replaces string1 with string2, in the range of lines specified by "line".

"Line" can be any of the parameters for a line described in Table 4-1. If no line is indicated, R operates on the line after the current line (marked by an *) through the end of the file.

- ▶ line is the specified line to the end of the file.
- ▶ line, line is the first specified line through the second specified line.
- ▶ ,line is the line after the current line through the specified line.
- ▶ .,line is the current line through the specified line.
- ▶ line,. is the range of lines from the specified line through the current line.

? prompts you to OK each string replacement or deletion. When EDLIN encounters a matching string of text, it displays the line and the prompt "O.K.?". If you press Y or Return, EDLIN makes the replacement or deletion and searches for the next matching string. If you press any key other than Y or Return, EDLIN leaves the string as is and searches for the next matching string.

string1 is the string of characters to be replaced. string2 is the string of characters to replace string1. CTRL-Z separates string1 from string2.

If you specify only one string of characters, R deletes all occurrences of the string within the specified range of lines.

In this example, the following file exists and is ready for editing:

```
1:REM This is a file to check a formatted disk
2:REM It is named NEWDISK.BAT
3:PAUSE Insert a formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

To replace all occurrences of "a" with "one" in lines 1 through 3, enter:

```
1,3Ra^Zone(cr)
```

The result is:

```
1:*REM This is one file to check one formonetted
  disk
2:It is nonemed NEWDISK.BAT
3:PAUSE Insert one formonetted disk in drive B
```

Some of these replacements produce nonsense words. To avoid unwanted substitutions, add the ? parameter:

```
1,3?Ra^Zone(cr)
```

Now each specified line is displayed, and you can confirm the changes. For example:

```
     1:*REM This is one file to check a formatted
       disk
O.K.?y
     1:*REM This is one file to check one
       formatted disk
O.K.?y
     1:*REM This is one file to check one
       formonetted disk
O.K.?n
     2:REM It is nonemed NEWDISK.BAT
O.K.?n
     3:PAUSE Insert one formatted disk in drive B
O.K.?y
```

After making the replacements you want, enter an L command to list the edited file:

```
*L(cr)

1:REM This is one file to check one formatted disk
2:REM It is named NEWDISK.BAT
3:*PAUSE Insert one formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

Search for a String (S)

[line][,line][?]Sstring(cr)

S searches for the string in the range of lines specified by line. It displays the first line containing the string. This line becomes the current line if the ? parameter is not specified.

"Line" can be any of the parameters for a line described in Table 4-1. The syntax for "line" is given in the Replace command description. If you do not specify a line, S operates on the line after the current line through the end of the file.

The ? prompts you to OK the displayed line as the current line. When EDLIN encounters a matching string of text, it displays the line and the prompt "O.K.?". If you press Y or Return, the displayed line becomes the current line. If you press any key other than Y or Return, EDLIN searches for the next matching string.

Assume the following file exists and is ready for editing:

```
1:REM This is a file to check a formatted disk
2:REM It is named NEWDISK.BAT
3:PAUSE Insert a formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

To search for the first "in" within lines 1 through 3, enter:

```
1,3Sin(cr)
```

The result is:

```
3:PAUSE Insert a formatted disk in drive B
```

To find "DIR" past line 3, specify:

3SDIR(cr)

The search starts at line 3 and continues to the end of the file. EDLIN displays the line containing the first "DIR":

```
4:DIR B:
```

To search the file for the first occurrence of "B", at the * prompt enter:

?SB(cr)

The result is:

```
    2:REM It is named NEWDISK.BAT
O.K.?_
```

If you respond Y, the search ends and you return to the * prompt. To continue the search, respond N.

Write Lines (W)

[n]W(cr)

W writes lines from memory to disk and makes room in memory for more lines to be appended (with the Append command) from disk to memory.

n represents the number of lines to be written.

If no parameter is given, lines are written to disk until memory is only 1/4 full. If a number is given, the specified number of lines is written to disk. As lines are written, subsequent lines are renumbered beginning with 1.

An A command following the W command appends lines from disk to any line remaining in memory.

MS-DOS Command Editing, Filtering, and Piping

This chapter describes how to enhance your control of MS-DOS commands. Chapter 5.1 describes the special MS-DOS editing keys and how you can use function keys to redisplay and edit parts of the previous command line (the template).

Chapters 5.2 and 5.3 demonstrate two major processes—filtering and piping data. These processes enable you to reorganize and output data between files or devices, and to use the output from one command as input to another command in the same command line.

5.1 Special MS-DOS Editing Keys

The way MS-DOS uses special editing keys differs from the way in which most operating systems handle command input. You do not need to type the same sequences of keys repeatedly because the last command line is automatically placed in a special storage area called a **template**.

By using the template and the special editing keys, you can take advantage of the following MS-DOS features:

1. You can repeat a command line instantly by pressing two keys.
2. If you make a mistake in the command line, you can edit it and retry the command without having to retype the entire command line.
3. By pressing a special editing key, you can edit and execute a command line that is similar to a preceding command line with a minimum of typing.

Figure 5-1 shows the relationship between the command line and the template.

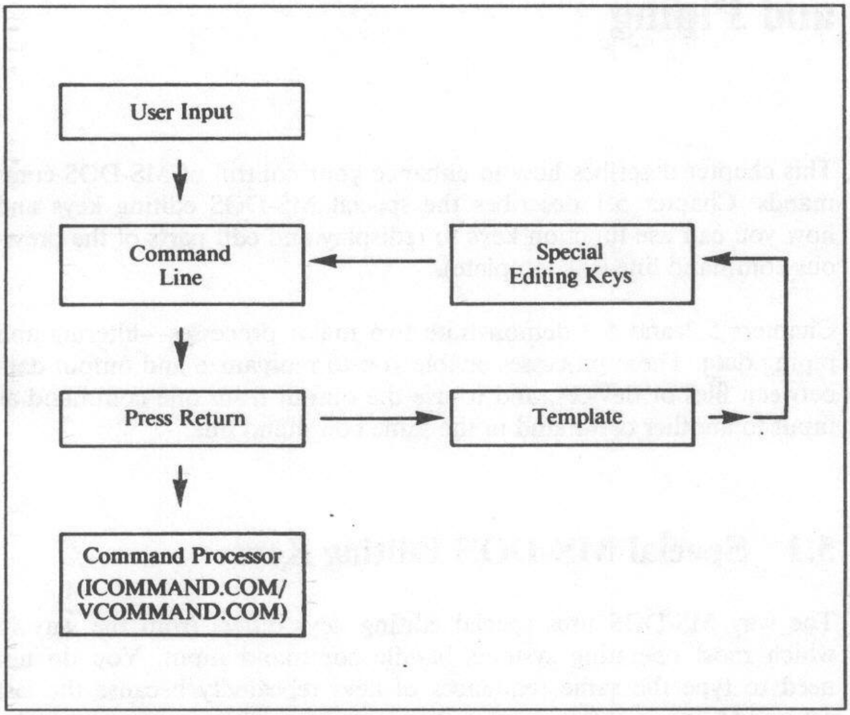


Figure 5-1: The Command Line and the Template

As shown in the figure, you type a command to MS-DOS on the command line. When you press the Return key, the command is automatically sent to the command processor for execution. At the same time, a copy of this command is sent to the template. Once a copy of the command is in the template, you can recall the command or modify it with the MS-DOS special editing keys.

Chapter 5.1.1 discusses the command-line editing functions available in I mode. Chapter 5.1.2 discusses the command-line editing functions available in V mode. Although many of the same functions exist in both modes, in many cases you press different keys to perform these functions.

For more information on the keyboard and how it behaves in I mode and V mode, see the *PlusPC User's Guide.*

5.1.1 Command-Line Editing in I Mode

This section summarizes what the command-line editing keys do in I mode. Table 5-1 lists the function keys associated with command-line editing in I mode. The examples following the table show you how these keys work.

Table 5-1: The I Mode Editing Functions

KEY	EDITING FUNCTION
F1 or →	Copies one character from the template to the command line.
F2	Copies characters up to the character specified in the template and puts these characters on the command line.
F3	Copies all remaining characters in the template to the command line.
DEL	Skips over (does not copy) a character in the template.
F4	Skips over (does not copy) the characters in the template up to the character specified.
ESC	Voids the current input; leaves the template unchanged.
INS	Enters/exits insert mode.
F5	Makes the new line the new template.
F6	Puts a CTRL-Z (1AH) end-of-file character in the new template.
BACKSPACE or ←	Backspaces and erases one character from the display line; leaves the template unchanged.

If you type the following command, MS-DOS displays information about the file PROG.COM on your screen:

```
dir prog.com(cr)
```

This command line is also saved in the template. To repeat the command, just press F3; the command is displayed on the screen. Then press Return. Notice that pressing the F3 key causes the contents of the template to be copied to the command line; pressing Return causes the command line to be sent to the command processor for execution.

If you want to display information about a file named PROG.ASM, you can use the contents of the template and type:

```
<F2>c(cr)
```

where <F2> represents function key 2. Pressing F2 and then C copies all characters from the template to the command line, up to but not including C. MS-DOS displays:

```
DIR PROG._
```

In the example above, the underline represents your cursor. Now type:

```
asm
```

The result is:

```
DIR PROG.ASM_
```

The command line DIR PROG.ASM is now in the template and is ready to be sent to the command processor for execution. To do this, press Return.

Now assume that you want to execute the following command:

type prog.asm

To do this, type:

type < INS > < F3 > (cr)

where < INS > represents the INS (Insert) key and < F3 > is function key 3. As you type, the characters are entered directly into the command line and overwrite corresponding characters in the template. This automatic replacement feature is turned off when you press the INS key. Thus, the characters "TYPE" and "PROG.ASM" replace the characters "DIR" in the template. To insert a space between "TYPE" and "PROG.ASM", you press INS and then the space bar. Finally, to copy the rest of the template to the command line, you press F3 and then Return. The command TYPE PROG.ASM is then processed by MS-DOS, and the template becomes TYPE PROG.ASM.

If you misspell TYPE as BYTE, you will receive an error message. But you can save the misspelled line before you press Return by creating a new template with the F5 key:

byte prog.asm < F5 >

You can now edit this erroneous command by typing:

t < F1 > p < F3 >

The F1 key (or the → key) copies a single character (in this case, y) from the template to the command line. Enter p, then press F3 to copy the rest of the template to the command line. The correct command line is displayed:

```
TYPE PROG.ASM
```

Or you can use the same template containing BYTE PROG.ASM and then use the DEL and INS keys to achieve the same result:

<DEL> <DEL> <F1> <INS> yp <F3>

To illustrate how the command line is affected as you type, examine the keys typed on the left; their effect on the command line is shown on the right:

KEY	DISPLAY	DESCRIPTION
<DEL>	—	Skips over 1st template character
<DEL>	—	Skips over 2nd template character
<F1>	T	Copies 3rd template character
<INS> YP	TYP	Inserts two characters
<F3>	TYPE PROG.ASM	Copies rest of template

Notice that DEL does not affect the command line. It affects the template by deleting the first character. Similarly, F4 deletes characters in the template, up to but not including a given character.

5

5.1.2 Command-Line Editing in V Mode

This section discusses the command-line editing key functions in V mode. These function keys, F1 through F7, are located on the top row of your keyboard.

Table 5-2 summarizes how the command-line editing function keys behave in V mode. The examples following the table familiarize you with how these keys work.

Table 5-2: The V Mode Editing Functions

KEY	EDITING FUNCTION
F1 or →	Enters or exits insert mode (toggle).
F2	Makes the new line the new template.
F3	Copies one character in the template to the command line.
F4	Skips over (does not copy) one character in the template.
F5	Skips over (does not copy) the characters in the template up to the character specified.
F6	Copies all the characters up to the character specified from the template to the command line.
F7	Copies all remaining characters in the template to the command line.
ALT-X	Cancels the line just entered; the template remains unchanged. Also, an ALT-X exits insert mode if it is on.

If you type the following command in V mode, MS-DOS displays information about the file PROG.COM on your screen:

```
dir prog.com(cr)
```

This command line is also saved in the template. To repeat the command, press F7. Regardless of the cursor position at the time, the whole command line displays on the screen. If you press Return, the command line goes to the command processor for execution.

If you want to display information about a file named PROG.ASM, you can use the contents of the template and press:

```
<F6>c(cr)
```

where <F6> represents function key 6. Typing F6 and then C copies all the characters from the template to the command line, up to but not including C.

MS-DOS displays:

```
DIR PROG._
```

In the example above, the underline represents your cursor. Now type:

asm

The result is:

```
DIR PROG.ASM_
```

The command line DIR PROG.ASM is now in the template and is ready to be sent to the command processor for execution. To do this, press Return.

Now assume that you want to execute the following command:

type prog.asm

To do this, type:

type <F1> (sp) <F7> (cr)

As you type, the characters are entered directly into the command line and overwrite corresponding characters in the template. Thus, the characters "TYPE" replace the characters "DIR" in the template. F1 turns on insert mode. To insert a space between "TYPE" and "PROG.ASM", press F1 and then the space bar. Finally, to copy the rest of the template to the command line, you press F7 and then Return. The command TYPE PROG.ASM is processed by MS-DOS, and the template becomes TYPE PROG.ASM.

If you misspell TYPE as BYTE, you will receive an error message. But you can save the misspelled line before you press Return by creating a new template with the F2 key:

byte prog.asm <F2>

You can edit this erroneous command by typing:

t<F3>p<F7>

The F3 key copies a single character (in this case, y) from the template to the command line. You then type p, and press F7 to copy the rest of the template. The correct command line is displayed:

```
TYPE PROG.ASM
```

The F4 key skips over one character in the template. The specified character does not appear in the new line. The cursor does not move; only the template is affected. For example, assume you mistype the following command:

chhdir myfiles(cr)

To correct this error, type ch and press F4. F4 skips over the second h. Then press F7 to copy the rest of the template:

ch<F4> <F7>

The result is:

```
CHDIR MYFILES
```

The F5 key skips over multiple characters in the template. If the template does not contain the specified character, nothing is skipped. Suppose you enter the following command:

```
cd myfiles
```

Now you want to change the command to RM MYFILES. Type F5 and the letter m. Next press F1 to enter insert mode, and type rm followed by a space. Press F7 to copy the rest of the command line:

```
<F5> m <F1> rm(sp) <F7>
```

The new command is displayed:

```
RM MYFILES
```

5.2 Filtering Data

MS-DOS can sort through the data in a text file and arrange it in a different order. This process is called **filtering**. These commands filter data:

- ▶ FIND searches for a constant string of alphanumeric characters.
- ▶ SORT rearranges the data in the order that you specify in the command. Order is predefined by the ASCII collating sequence. See Appendix A for an ASCII conversion table.
- ▶ MORE displays one screen of data at a time; press the Return key to display another screen of data. See Chapter 6.4 for information on the MORE command.

You can use these commands with any MS-DOS files, such as VBASICA files. The examples given in this section use the following data file, PHNLST.DAT, containing four records of names and phone numbers:

```
Washington, George 301 328-4370
Kennedy, John 617 736-2867
Lee, Robert 703 532-3370
Lincoln, Abraham 309 473-2800
```

You can filter the data by area code with the FIND and SORT commands. In the following example, FIND searches for a numeric string of characters, the 309 area code:

```
find "309" b:phnlst.dat(cr)
```

MS-DOS displays:

```
----------b:phnlst.dat
Lincoln, Abraham 309 473-2800
```

The SORT command can arrange data alphabetically or the reverse, Z through A. The following command tells MS-DOS to arrange the PHNLST.DAT data records alphabetically and to output the sorted data to the screen:

```
sort < b:phnlst.dat > con(cr)
```

The < and > symbols redirect input and output. The < in the previous command tells MS-DOS that the file PHNLST.DAT on drive B is the input for the SORT command. The > symbol tells MS-DOS to send the output (the sorted data) to the display unit (CON).

The preceding command displays this sorted data:

```
Kennedy, John  617 736-2867
Lee, Robert 703 532-3370
Lincoln, Abraham 309 473-2800
Washington, George 301 328-4370
```

You can also tell MS-DOS to create a new file and send sorted data to the new file. For example, if you name the file ALPHA.NUM in place of CON in the previous example, MS-DOS sends the alphabetized file records to ALPHA.NUM on drive B:

```
sort < b:phnlst.dat > b:alpha.num(cr)
```

5.3 Piping Data

To **pipe** data means to enter more than one command on a command line, separating commands with a vertical bar (|), the pipe separator. By piping commands you can use the output from one command as the input for another command. You can combine piping with filtering to rearrange existing files into new sorted files.

You can:

- ▶ Filter data from an existing data file and pipe it to a new file or other output source, such as the screen (CON) or printer (PRN).
- ▶ Filter columnar data, such as a directory, by column number (the screen column where file size, date, or time is listed).
- ▶ Use batch processing to make filtering and piping data even easier.

5.3.1 Filtering and Piping Data Files

Using PHNLST.DAT, the sample file shown in Chapter 5.2, you can filter the data by area code. You can then pipe the filtered area codes to the AREA.COD file by specifying:

```
find "301" b:phnlst.dat | sort > b:area.cod(cr)
```

The | pipe symbol tells MS-DOS to take the output from the command on the left side of the bar and use it as input for the command on the right side of the bar. The > symbol in the command sends all the records containing the 301 area code to a new file named AREA.COD.

The following command uses the >> symbol to append all the records containing the 309 area code to the end of the AREA.COD file:

```
find "309" b:phnlst.dat | sort >> b:area.cod(cr)
```

To verify that the sorted records are in AREA.COD, specify:

```
type b:area.cod(cr)
```

If you enter:

```
type b:phnlst.dat | sort > prn(cr)
```

MS-DOS gets the output of the TYPE B:PHNLST.DAT command and uses it as input for the SORT command. The SORT command sorts the file alphabetically and sends the sorted data to the printer.

5.3.2 Filtering and Piping Columnar Data

Using the SORT command, you can pipe and sort columns of data. For example, if you type:

```
dir | sort /+16 | more(cr)
```

MS-DOS pipes the output of the DIR command as input to the SORT filter command. The sorting starts in column 16 of the directory display (file size). MS-DOS sends the directory, sorted by file size, to the display unit. MORE displays the directory one screen at a time with the message --MORE-- at the bottom of the screen. You can display the next screenful of directory information by pressing Return.

5.3.3 Filtering and Piping Batch Files

5

You can use filtering and piping in batch files. The next example creates a batch file PHONES.BAT that searches for a value (the %1 parameter) in the PHNLST.DAT file, sorts the records containing that value, and displays those records onscreen:

```
copy con phones.bat(cr)
find "%1" phnlst.dat | sort >(cr)
^Z(cr)
```

You can use PHONES.BAT to search for any area code you want. Type PHONES and the area code you want to have sorted from the PHNLST.DAT file, such as:

```
phones "301"(cr)
```

All the records from PHNLST.DAT with area code 301 are displayed.

MS-DOS Commands

This chapter presents an overview of MS-DOS commands. Chapter 6.1 lists and describes the two types of MS-DOS commands (internal and external) as well as the batch commands (internal commands used in batch files). Chapter 6.2 describes the notation used in the syntax of commands in this manual.

Chapter 6.3 describes the special commands you use to alter the system configuration contained in the ICONFIG.SYS/VCONFIG.SYS files. Finally, Chapter 6.4 presents the syntax of all the MS-DOS commands (excluding EDLIN and the system configuration commands). Commands are listed alphabetically and examples are included for most commands.

6.1 Overview of Commands

You use the MS-DOS commands to set up and manage your files and directories, the diskettes that contain your files, and the devices attached to your computer. This overview describes some of the MS-DOS commands and what you can do with them. For a complete listing of the commands, see Chapter 6.4.

You use the MS-DOS file commands to:

- ▶ Copy, delete, rename, display, or verify files (COPY, DEL, REN, TYPE, and VERIFY)
- ▶ Copy the operating system files from one drive to a VICTOR format diskette or fixed disk volume (RETROSYS)
- ▶ Change the file format or rearrange the file contents (SORT and FIND)

- ▶ Queue a list of files for printing (PRINT)
- ▶ Create batch files and control their execution (COPY CON, ECHO, SHIFT, and so on)

You use the MS-DOS directory commands to:

- ▶ Display the directory of a diskette (DIR)
- ▶ Create or remove directories (MKDIR and RMDIR)
- ▶ Change from one subdirectory to another, or check your position in the directory hierarchy (CHDIR)

You use the MS-DOS disk commands to:

- ▶ Prepare blank diskettes for MS-DOS use (FORMAT).
- ▶ Copy the contents of one diskette to another (DISKCOPY).
- ▶ Check the amount of space on a diskette, and check a directory for errors in file storage (CHKDSK)
- ▶ Recover data from a bad sector on a disk (RECOVER)

The MS-DOS device commands (CLS, CTTY, CLST, and CAUX) control the CRT, your printer, and your disk drives. These commands display information about your diskettes and disk drives. They can also change devices, such as changing output from the screen to the printer.

The system configuration commands (such as BREAK, BUFFERS, and DEVICE) enable you to control parts of the operating environment. All of these commands are used in the ICONFIG.SYS and VCONFIG.SYS files, which set up the environment when MS-DOS loads into memory. The BREAK command can also be used interactively. See Chapter 6.3 for a description of the configuration commands.

Other MS-DOS commands affect the way the operating system works. For example, there are commands to set up constants in the operating environment and commands to change the system command prompt.

6.1.1 Internal and External Commands

There are two types of MS-DOS commands: internal and external. Internal commands are a set of commands that exist within MS-DOS. External commands are stored on disk as separate files. External commands have either a .COM or .EXE file extension. External commands are also known as utility programs.

The command processor program interprets and processes all the MS-DOS commands, both internal and external. The command processor is in the system files ICOMMAND.COM and VCOMMAND.COM contained on the MS-DOS system diskette. When you load MS-DOS, ICOMMAND.COM, VCOMMAND.COM, or any command processor file named in ICONFIG.SYS/VCONFIG.SYS is automatically loaded into memory.

The internal commands are part of the ICOMMAND.COM/ VCOMMAND.COM files. External commands, on the other hand, must be individually loaded into memory by ICOMMAND.COM or VCOMMAND.COM each time you use that command. The internal commands remain in memory as long as ICOMMAND.COM or VCOMMAND.COM is in memory.

The MS-DOS command interface is illustrated in Figure 6-1.

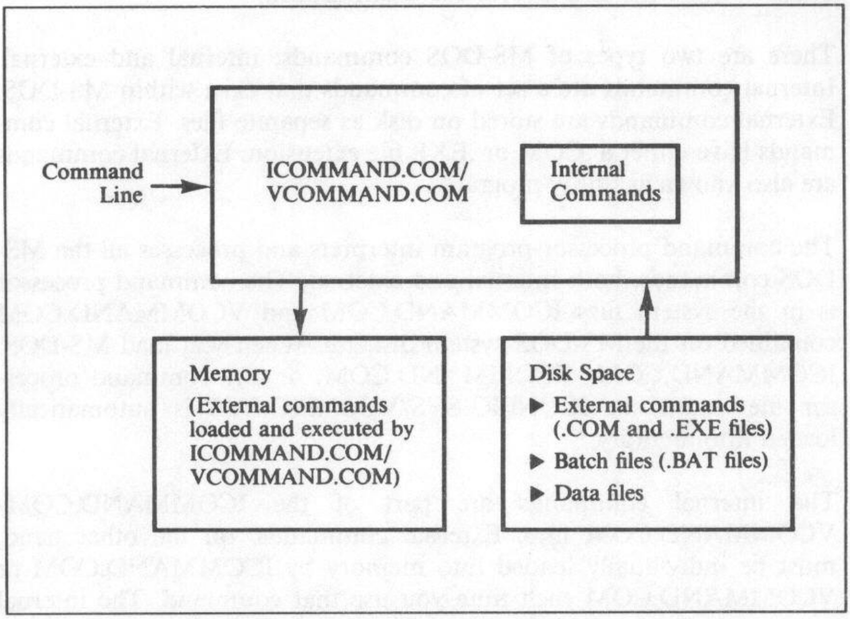

Command Line
ICOMMAND.COM/ VCOMMAND.COM
Internal Commands
Memory
(External commands loaded and executed by ICOMMAND.COM/ VCOMMAND.COM)
Disk Space
▶ External commands (.COM and .EXE files)
▶ Batch files (.BAT files)
▶ Data files

Figure 6-1: MS-DOS Command Interface

Internal Commands

The internal commands are:

BREAK	CTTY	HISTORY	SET
CAUX	DATE	MKDIR	TIME
CHDIR	DEL	PATH	TYPE
CLS	DIR	PROMPT	VER
CLST	ERASE	REN	VERIFY
COPY	EXIT	RMDIR	VOL

These internal commands are not displayed in the directory. When you enter an internal command, it executes immediately whenever ICOMMAND.COM/VCOMMAND.COM is in memory.

Some internal commands can use the pathing system described in Chapter 3 to obtain files from the MS-DOS subdirectory system. These commands are COPY, DEL, DIR, CHDIR, ERASE, MKDIR, RMDIR, TYPE, and PATH.

External Commands

The external commands are:

ASSIGN	EXESIZE	MODCON	SEARCH
CHKDSK	FGREP	MODE	SORT
COMP	FILCOM	MORE	TAIL
CONCAT	FIND	MV	TREE
DISKCOMP	FORMAT	PRINT	VCOMMAND
DISKCOPY	GRAPHICS	RECOVER	WC
EDLIN	ICOMMAND	RETROSYS	
EXE2BIN	LS	SDCOPY	

The command descriptions in Chapter 6.4 include all the external commands except EDLIN (see Chapter 4).

Like all programs, the external commands are stored in command files on disk. You can identify them by the extension .COM or .EXE. You should keep all the MS-DOS commands in the subdirectory structure provided for you on your PlusPC distribution diskette.

Before you can successfully issue an external command, MS-DOS has to know where to locate the command file. Use either the CHDIR command to change to the directory where you keep your command files, or use the PATH command to specify a path to that directory. When you enter the command, MS-DOS reads the command from that directory. If MS-DOS cannot find the command file, you see this message:

```
Bad command or filename
```

6.1.2 Batch Commands

Batch commands are used only in batch files (see Chapter 2). Because the batch commands are internal commands, ICOMMAND.COM/ VCOMMAND.COM must be loaded in main memory whenever you run a batch file that uses any of these commands:

ECHO	IF–EXIST
FOR–IN–DO	PAUSE
GOTO	REM
	SHIFT

6.2 Command Syntax and Notation

Descriptions of all the MS-DOS commands are listed alphabetically in Chapter 6.4. In each description, the syntax or format for each command is given first. Then the elements of the command are defined, and examples are presented to show how to use the commands. In examples, the entire command line is lowercase except for elements that require uppercase, such as proper nouns or command switches that are not valid in lowercase.

The notation used for command syntax is as follows.

COMMAND NAMES
: are shown in uppercase. You can enter command names in upper- or lowercase, however.

LITERAL PARAMETERS
: are shown in uppercase. These are parts of a command that must be entered literally if you want to include them. You can enter the parameters in upper- or lowercase, except as noted in the command descriptions.

variable parameters

are shown in lowercase. You can enter command parameters in upper- or lowercase.

[ ]

enclose optional parameters.

...

is an ellipsis indicating that you can repeat the preceding parameter.

filename.ext, drivename

represent the type of entry you must make. For example, when you see the word filename, you must enter the name of a specific file (with or without wild-card characters).

path

is the full or relative path through the subdirectory hierarchy to the subdirectory or file named in the command. In this chapter, the syntax for paths is:

```
[drivename:][[\]directoryname[\directoryname...]]
```

If the file or subdirectory you want is in the current directory on the default drive, you can omit the directory and drive names. The first \ in a path always refers to the root directory. Each additional \directoryname refers to a lower level subdirectory. Path syntax is fully described in Chapter 3.

/

indicates that the parameter (such as /C) is a switch. A switch causes MS-DOS to process the command as altered by the specified switch. When you enter a switch in the command line, separate it from commands and other switches with a space.

|

indicates that data is to be piped (see Chapter 5). The output generated by the command on the left side of the bar is sent as input to the command on the right side of the bar. Separate the | from other command elements with a space.

<

indicates that the input to the preceding command is to come from a source other than the keyboard. Enter a space before and after the <.

>

indicates that the output from the preceding command is to be sent to a file or device (such as the printer). Enter a space before and after the >.

>>

indicates that the output from the command is to be sent to the end of the file named in the command. Separate the >> from other command elements with a space, but do not type a space between the >s.

(cr)

indicates the Return key.

(sp)

indicates the Space bar.

^

indicates the CTRL (Control) key. A CTRL-key sequence (pressing CTRL and another key) is shown as CTRL- and the letter or number, such as CTRL-C.

You must include all punctuation (commas, colons, or question marks), symbols \, |, <, and >>, and blank spaces where shown, except for [ ], which indicate optional parameters, and the ellipsis. The ellipsis indicates that a parameter can be repeated.

NOTE: Because the symbols \, |, <, and > are literal parts of MS-DOS commands, they cannot be used in a filename. If you have any existing files that have \, |, <, or > in their filenames, you must rename the files before you can use them with MS-DOS version 2.1 or later. Use a previous version of MS-DOS to rename the files.

6.3 System Configuration Commands

In many cases, there are installation-specific settings for MS-DOS that need to be configured at system startup. An example of this is a standard device driver, such as an online printer.

The MS-DOS configuration file (ICONFIG.SYS/VCONFIG.SYS) allows you to configure your system. With this file, you can add device drivers to your system at startup. The configuration file is an ASCII file that has certain commands for MS-DOS startup (boot). The boot process is as follows:

1. The disk boot sector is read. This contains enough code to read MS-DOS code and the installation's BIOS (machine-dependent code).
2. The MS-DOS code and BIOS are read.
3. A variety of BIOS initializations are done.
4. A system initialization routine reads the configuration file, if it exists, to perform device installation and other user options. Finally, it executes the command interpreter, finishing the MS-DOS boot process.

Chapter 6.3.1 describes the commands that you can use in the configuration file. You do not need to change this file unless you want to alter the operating environment to fit your needs.

When MS-DOS loads into memory, it checks the root directory for a system configuration file named ICONFIG.SYS/VCONFIG.SYS. This file contains system initialization routines that tell MS-DOS:

- ▶ Whether a reference to a device requires a pathname (AVAILDEV)
- ▶ When to allow CTRL-C to interrupt operations (BREAK)
- ▶ The number of buffers to allocate for input and output data (BUFFERS)

- The standards associated with particular countries (COUNTRY)
- The device drivers to be installed (DEVICE)
- The number of files that can be open at the same time (FILES)
- The name of the command processor to be loaded (SHELL)
- The symbol that indicates a command parameter (SWITCHAR)

To change the ICONFIG.SYS or VCONFIG.SYS file, use any text editor that operates under MS-DOS, such as EDLIN. You can also create a new configuration file with the COPY command. The changes you make to the file become effective the next time you load MS-DOS.

6.3.1 Descriptions of Configuration Commands

This section presents the syntax for the system configuration commands that can appear in ICONFIG.SYS/VCONFIG.SYS.

AVAILDEV

```
AVAILDEV = TRUE(cr)
```

or

```
AVAILDEV = FALSE(cr)
```

The default is AVAILDEV = TRUE. This command controls whether device names (such as CON and LST) refer only to devices or can be used as filenames. The device files are described in Chapter 2.2. The default setting means MS-DOS reads both the following names as the device name LST. LST is used to represent any device name:

```
\DEV\LST
LST
```

If AVAILDEV = FALSE, MS-DOS reads only \DEV\LST as the device name LST. This means that LST by itself refers to a file in the current directory with the same name as the device LST. Of course, the file might or might not exist.

BREAK

```
BREAK = ON(cr)
```

or

```
BREAK = OFF(cr)
```

The default, BREAK = OFF, tells MS-DOS to check for a CTRL-C or CTRL-BREAK interrupt only during screen, keyboard, printer, or asynchronous communication adapter operations. BREAK = ON tells MS-DOS to check for a CTRL-C or CTRL-BREAK interrupt during all operations, including whenever MS-DOS is processing a program. You can also change the setting by issuing the interactive BREAK command, described in Chapter 6.4. If you set BREAK = ON, you can abort programs by typing a CTRL-C or CTRL-BREAK.

BUFFERS

```
BUFFERS = xx(cr)
```

where xx is a number between 1 and 99. The default value is 3. A buffer is the amount of memory allocated for the reading of data from disk and the writing of data to disk. If your application does sequential access to the data on disk, you can use a small number of buffers. If you are using a database with random access, 10 to 20 buffers is adequate. A larger number of buffers can speed disk access. If the number of buffers gets too large, however, you use more memory and slow operations.

COUNTRY

```
COUNTRY = nn(cr)
```

where nn is a number between 1 and 99. The COUNTRY command selects configuration standards according to the given country code. Some of the standards affected are the currency separator and the format for the DATE command (such as listing the month first or listing the day first).

The numbers you can use with COUNTRY are the country's telephone code, for example:

1	USA
33	France
44	United Kingdom
46	Sweden
49	Germany
81	Japan

DEVICE

DEVICE = [path]filename.ext[parameters](cr)

This command installs the device driver from the specified file. The default is the device drivers for the standard screen, keyboard, printer, auxiliary device, floppy disk drive, and fixed disk. DEVICE also lets you include device drivers that you have written. For each device driver, you must include a DEVICE command in the ICONFIG.SYS/ VCONFIG.SYS file.

path
  is the path to the device driver file, including drive and subdirectory names where necessary.

filename.ext
  is the name of the file containing the device driver.

parameters
  is a list of initial settings for a device driver. Use the MODE command to set serial ports. See MODE in this chapter for more information.

The default device drivers for your input/output ports are:

▶ AUX—for serial port 1, also known as COM1.

▶ COM2—for serial port 2.

▶ PRN—for the parallel port.

FILES

FILES = xx(cr)

where xx is a number between 1 and 99. The default value is 8. This command specifies the number of files that can be opened concurrently. For each additional file above 8, the size of the resident portion of MS-DOS increases by 39 bytes per file. The amount of memory available for the application program is reduced by the number of bytes allocated to the number of opened files. **Note:** Specifying a value less than 8 might prevent some system operations, such as filtering and piping (see Chapter 5.2).

SHELL

SHELL = [path]filename.ext [comspec path][/P](cr)

SHELL names the command processor to be loaded when MS-DOS starts. The default is ICOMMAND.COM/VCOMMAND.COM. ICOMMAND.COM is for I mode; VCOMMAND.COM is for V mode. Do not alter these files unless you are a systems programmer who has created a command processor for your own applications. If you replace ICOMMAND.COM or VCOMMAND.COM with your own command processor, you must provide for the hex 22, 23, and 24 interrupts, the internal commands, the batch processor, and the EXEC function call.

path
: is the path through the directory hierarchy to the file containing the command processor. Include the drive name and subdirectory names where necessary.

filename.ext
: is the name of the file containing the command processor.

/P

tells the command processor that it is the first program to be run by MS-DOS. The P must be preceded by the correct symbol for command parameters (see SWITCHAR).

comspec path

is a path through the directory system. It is appended to the name of the command processor and SET into the MS-DOS environment as "COMSPEC". The operating system uses COMSPEC to locate the command processor. You can display the current COMSPEC path with the SET command. If omitted from the SHELL command in ICONFIG.SYS/VCONFIG.SYS, the COMSPEC path is the root directory on the default drive.

The following example uses VCOMMAND within a SHELL command line in a VCONFIG.SYS file:.

```
shell = a:\bin\vcommand.com(cr)
```

This SHELL command tells MS-DOS that the command processor is the file VCOMMAND.COM on drive A in subdirectory BIN.

SWITCHAR

```
SWITCHAR = x(cr)
```

where x can be any alphanumeric character. The default switch character is /. This command specifies the symbol that indicates command switches to MS-DOS. For example, to add the V switch to the COPY command, you would type /V.

By default, the MS-DOS path-separator character is the backslash (\), while the slash (/) is the switch character. If you set a different switch character with the SWITCHAR command, the path separator becomes the slash (/).

For example, if VCONFIG.SYS contains this command:

```
switchar = -
```

you must use the hyphen to introduce command switches (such as -V), and you must use the slash to separate parts of pathnames (such as /MGR1/TOM).

6.3.2 Sample Configuration File

A typical configuration file might look like this:

```
break = on
buffers = 10
files = 10
switchar = /
shell = a:\bin\vcommand.com
```

This VCONFIG.SYS file does the following:

- ▶ Sets the CTRL-C interrupt on
- ▶ Adds 10 additional sector buffers to the system list
- ▶ Allows 10 files to be open at once
- ▶ Defines the syntax notation for command parameters as a slash
- ▶ Begins execution of the shell (the top-level command processor) from VCOMMAND.COM in the subdirectory BIN on drive A

6.4 Command Descriptions

This section presents descriptions of the syntax and examples of usage for the MS-DOS commands. Each command is listed alphabetically.

ASSIGN

ASSIGN [x = y...](cr)

ASSIGN is an external command that instructs MS-DOS to use a drive other than the one requested. In effect, physical drive X is converted to logical drive Y. Note that when you use this command, you do not include a colon after the drive name.

x

is the physical drive.

y

is the logical drive assignment.

When you use the ASSIGN command without any parameters, all drive assignments are reset to normal.

With the ASSIGN command, you can use applications programs designed to use only drives A and B. For example, an applications program that offers only A and B as choices can also be used with drive C.

The ASSIGN command should be used only in such applications programs. It must never be used with the PRINT command or in normal MS-DOS operations because this can hide device-type commands and programs that require actual disk operation. DISKCOPY and DISKCOMP ignore all drive reassignments.

When you enter the following command, procedures previously assigned to drive A are performed on drive B. For example, if you enter this command and then a directory command, MS-DOS displays a directory listing of the files located in drive B.

```
assign a = b(cr)
```

The following command line causes procedures assigned either to drive A or to drive B to be performed on drive C:

assign a = c b = c(cr)

If you specify ASSIGN without any parameters, all previous assignments are undone; physical drive A is logical drive A, physical drive B is logical drive B, and so on:

assign(cr)

CTRL-C Interrupt (BREAK)

BREAK [ON](cr)

or

BREAK [OFF](cr)

BREAK is an internal command that lets you use CTRL-C as a system interrupt (BREAK ON) or without the interrupt effect (BREAK OFF) when MS-DOS is processing a program. When BREAK is ON, typing CTRL-C interrupts the current operation and redisplays the system prompt. Enter BREAK without parameters to check the status of the interrupt function. MS-DOS displays the following, where *xx* is "on" or "off."

```
BREAK is xx
```

ON tells MS-DOS to allow a CTRL-C interrupt at any time during the processing of a program. OFF tells MS-DOS to allow a CTRL-C interrupt only during the operation of the keyboard, printer, or an auxiliary device. The default is OFF.

Change Directory (CHDIR)

CHDIR [..] [path] [/A](cr)

CHDIR is an internal command that changes your position in the directory hierarchy or displays the name of the current subdirectory. You can abbreviate the command name to CD. CHDIR is also described in Chapter 3.

You use this command to:

▶ Move to a subdirectory from any other directory.

▶ Move to a directory level above this one.

▶ Return from a subdirectory to the root directory.

▶ Display the name of the directory you are using.

▶ Display the names of the working directories on all the drives in the system.

CHDIR without parameters displays the directory you are currently using. For example, if you are in a directory called CMD in the default drive and specify:

chdir(cr)

MS-DOS displays:

```
A:\cmd
```

..

Moves you to the directory above the one you are currently using. You can move up two directory levels by specifying:

```
cd ..\..(cr)
```

/A

displays the name of the current directory on each drive. Entering CHDIR /A tells you the paths to the default directories. This CHDIR command is useful when you are copying files from one drive to another. When you enter the COPY command, you can omit the paths for files you are copying from or to the default directories.

path

is the path through the directory system to the subdirectory level you want. Each directory name is the name of a directory created with the Make Directory (MKDIR) command. For example, the following command looks for the EDFILES subdirectory three levels below the root directory:

```
chdir \mgr1.nam\cmd\edfiles(cr)
```

If you specify a directory name without the \ parameter, MS-DOS looks for the directory name in the current directory. In the previous example, if you are in the CMD subdirectory and you want to access EDFILES, you can enter:

```
chdir edfiles(cr)
```

instead of the path, \MGR1.NAM\CMD\EDFILES. MS-DOS looks in the current directory, CMD, for the EDFILES subdirectory.

Entering \ without a directory name refers to the directory at the top of the directory hierarchy, the root directory. This CHDIR command returns you to the root directory:

```
chdir \(cr)
```

Check Disk (CHKDSK)

CHKDSK [drivename:][filename.ext][> filename.ext] [/F] [/V](cr)

CHKDSK.COM is an external command that scans the directory hierarchy of the diskette in the specified drive for consistency. If inconsistencies are found, MS-DOS displays error messages and a status report.

If you enter CHKDSK without parameters and if the diskette in the default drive contains errors, MS-DOS responds:

```
Errors found, F parameter not specified.
Corrections will not be written to disk.
n lost clusters found in n chains
Convert lost chains to files (y/n)?
```

where the first *n* is a number.

If you respond with Y, MS-DOS tries to convert the lost chains to files, but cannot because you did not use the /F parameter in the command line. If you respond with N, MS-DOS displays the amount of memory that would be freed.

drivename
  is the name of the drive containing the disk that you want to check.

filename.ext

is the name of the file you want to check. Use wild-card characters to refer to more than one file. For example, *.* reports extents (contiguous space on disk reserved for a file) for all files in the current directory after a normal consistency check is made. You can copy and rename files with many extents to restore them to a contiguous state.

> filename.ext

redirects output from the CHKDSK command to the specified file.

/F

automatically reports to you when:

- ▶ There is an allocation error. CHKDSK fixes the error by adjusting the size.
- ▶ There is a disk error during the reading of the File Allocation Table (FAT).
- ▶ There is a disk error during the writing of the File Allocation Table (FAT).
- ▶ The file for the specified filename contains noncontiguous blocks.
- ▶ The first cluster (file attributes, password, allocation, extent, and statistical information) number is invalid and the file entry is truncated. CHKDSK fixes the error by releasing the allocation units assigned to that file.
- ▶ A Change Directory command for the specified filename cannot be done because a subdirectory cannot be processed.

You must correct the error when CHKDSK encounters:

- ▶ An incorrect MS-DOS version. You cannot run this version of CHKDSK on MS-DOS versions lower than 2.1.
- ▶ Insufficient memory. Processing cannot continue. There is not enough memory in the processor unit for CHKDSK to process the disk.

- ▶ An invalid current directory. Processing cannot continue. Restart the system and reissue the CHKDSK command.
- ▶ A Change Directory command that cannot be made to the root directory. Processing cannot continue. The disk is a bad disk. Restart MS-DOS and issue the RECOVER command.
- ▶ Cross-linked files. The same data block is allocated to more than one file. Use the COPY command to copy the files you want to keep. Check the copies for accuracy, then delete the files that are cross-linked.
- ▶ A non-MS-DOS disk. You are prompted to respond yes (Y) or no (N) if you want CHKDSK to continue processing.
- ▶ Insufficient room in the root directory. You are prompted to erase the files in the root directory and reissue the CHKDSK command.
- ▶ An unrecoverable error in the directory. You are prompted to respond yes (Y) or no (N) to convert the directory to a file. If you respond Y, you can fix the directory yourself or delete it.
- ▶ Lost allocation units. Lost allocation units are often called "orphans." Usually orphan allocation units occur when you do not close a file correctly (created, written to, but not closed). You can either free the orphan allocation units or recover them. A CHKDSK recover operation writes orphan allocation units to files in the root directory of the disk. If there is not enough room on the disk to write all the orphan files, delete some files, and run CHKDSK again.

/V

displays a trace (a record of the series of events as they occur) of the files and directories during the CHKDSK processing.

The following example checks the diskette in drive **B** for consistency and issues a status report:

```
A>chkdsk b:(cr)

  620544 bytes total disk space
   63488 bytes in 2 hidden files
    2048 bytes in 1 directories
  237568 bytes in 16 user files
  317440 bytes available on disk

  472976 bytes total memory
  458576 bytes free

A>_
```

Clear Screen (CLS)

CLS(cr)

CLS is an internal command that clears the CRT screen and redisplays the system prompt.

Compare Files (COMP)

```
COMP [drivename:][path][filename[.ext]]
   [drivename:][path][filename[.ext]](cr)
```

COMP is an external command that compares the contents of one set of files to the contents of another set of files.

drivename:path\filename.ext
: The first parameter is the primary file, and the second is the secondary file. These files can be on the same drive or on different drives, in the same directory, or in different directories.

Wild-card characters can be used in either of the filenames and extensions. If you specify only the drivename and the path, and do not specify a filename, MS-DOS assumes a filename of *.*.

To compare identical files in two different directories on two different drives, enter a command similar to this:

```
comp a:\dir1\*.c b:\dir2\dir3(cr)
```

Each file located in the directory DIR1 on drive A and having a .C extension is compared to a file with a matching filename and extension and located on drive B in the path DIR2\DIR3.

As each file is compared, its pathname and filename are displayed.

If the files being compared have different lengths, MS-DOS displays the following message, and no comparison takes place:

```
Files are of different sizes.
```

If information in the two files being compared is found not to match, the differences are displayed as follows:

```
Compare error at offset nnnnnn
File 1 - xx
File 2 - xx
```

where *nnnnnn* is the number of bytes from the beginning of the file to the byte where the files differ.

Up to ten compare errors can be reported for any given file. If there are more than ten errors, MS-DOS assumes that further comparison is useless, and it goes on to the next file.

If no compare errors are found, the following message is displayed:

```
Files compare OK
```

Certain applications programs create directories that always record file size in multiples of 128. Therefore, the number of information bytes in a file might be somewhat less than is indicated in the directory. In such cases, the information bytes in the two files might compare, but the COMP command might find differences in the non-usable (filler) portion of the file that follows the information bytes. If this occurs, MS-DOS displays the following message:

```
EOF mark not found.
```

Concatenate and Output Files (CONCAT)

```
CONCAT [files or directories](cr)
```

CONCAT.EXE is an external command that concatenates (combines) a list of files and displays their contents. The concatenated files are sent to the standard output device, the console, but are not saved on disk unless you include the > redirection symbol to send the output to a disk file (shown in the first example that follows).

files or directories
: is a list of subdirectory names and/or filenames to be operated on. If you do not specify files or directories, input is taken from the keyboard or from redirected input.

For example, the following command concatenates the TOD.ASM and ATC.ASM files in the current directory. The > symbol sends the combined files to the file BACKUP.

```
concat tod.asm atc.asm > backup(cr)
```

You can also use CONCAT to create files. This command takes any following keyboard input (up to CTRL-Z Return) and outputs it to the disk file JUNK:

```
concat > junk(cr)
```

The following command displays the contents of all files in the \TEMP subdirectory on the default drive and all files with extension .PLM in the current directory:

```
concat \temp *.plm(cr)
```

Copy Files (COPY)

```
COPY [path]filename.ext [/A] [/B] [+ [path]filename.ext [/A] [/B]...]
   [path][filename.ext] [/A] [/B] [/V](cr)
```

COPY is an internal command that copies one or more files from a source file to a destination file. The source file is the file you want to copy. The destination file is the file to which you want to copy your source file. If you do not specify a destination drive, your file is copied to the current or default drive.

path
: is the path through the directory hierarchy to the source or destination file. The path includes the drive name and subdirectory names as necessary.

filename.ext
: is the name defined when the file was created. It can also be the name of a new file created during the COPY operation.

6

To copy a file to the default drive, specify the drive containing the source file, then the filename. After this file copy, the copy in the default drive has the same name as the source file. For example, this command copies the file from drive B to the default drive (drive A) and gives it the same name:

```
copy b:filename(cr)
```

The copy is made to the current directory of the default drive.

To copy the file from the default drive to another drive, specify the command name, the name of the file to be copied, and the name of the drive. For example, this command copies the file in the default drive to drive B and gives it the same name:

```
copy filename b:(cr)
```

Using only filename parameters, you can:

- ▶ Copy a source file from another drive to a destination file on the default drive and keep the same filename:

```
copy b:chkdisk.bat(cr)
```

- ▶ Copy a source file from the default drive to a destination file with the same name on a different drive:

```
copy chkdisk.bat b:(cr)
```

- ▶ Copy a source file on the default drive to a destination file on the same drive and rename the file:

```
copy chkdisk.bat newfile(cr)
```

  The CHKDISK.BAT file is copied on the default drive and renamed NEWFILE.

- ▶ Copy a source file from the default drive to another drive and rename the file:

```
copy chkdisk.bat b:newfile(cr)
```

- ▶ Copy a source file from a specified drive to the same drive and rename it:

```
copy b:chkdisk.bat newfile(cr)
```

  The file CHKDISK.BAT on drive B is copied to drive B and renamed NEWFILE.

+

concatenates files (links them together). MS-DOS adds a file or files to the end of the first specified file and puts the result in either the first file or the destination file that you name. For example, this command adds the file called PLUS.FIL to the end of ONE.FIL and puts the result in ONE.FIL:

```
copy one.fil + plus.fil(cr)
```

The next two examples put the result of the concatenated files into another file:

```
copy one.fil + plus.fil big.fil(cr)
```

The contents of the file PLUS.FIL are added to the end of ONE.FIL and the result is renamed BIG.FIL.

```
copy one.fil + plus.fil + b:next.fil big.fil(cr)
```

The contents of the file NEXT.FIL on drive B are added to the end of the file PLUS.FIL on the default drive. The resulting file is added to the end of ONE.FIL on the default drive. Finally, the destination file is called BIG.FIL.

You can concatenate files using wild-card characters:

```
copy *.lst + *.ref combin.fil(cr)
```

This command searches for and combines all the files with extensions .LST and .REF. The result of the concatenation is put into COMBIN.FIL. FILE1.LST is combined with FILE1.REF and the result is put in COMBIN.FIL.

The following command combines FILE1.LST with FILE1.REF to form FILE1.FIL. ABC.LST combines with ABC.REF to form ABC.FIL, and so on.

```
copy *.lst + *.ref *.fil(cr)
```

This command can produce an error if ALL.LST already exists:

```
copy *.lst + all.lst(cr)
```

As each source filename is found, MS-DOS compares the name with the destination filename. If the names are the same, the source file is skipped and MS-DOS displays:

```
Content of destination lost before copy
```

Further concatenation continues normally.

You can, however, add all the files with extension .LST to the file ALL.LST:

copy all.lst + *.lst(cr)

/A

specified with the source file causes the file to be processed as an ASCII (text) file. The file is copied to the first end-of-file character (CTRL-Z for I mode, or ALT-Z for V mode). The CTRL-Z and the rest of the file are not copied. For example:

copy file.txt /a memo.txt(cr)

If there is a CTRL-Z within FILE.TXT, the CTRL-Z is removed when FILE.TXT is copied to MEMO.TXT. MEMO.TXT contains only one CTRL-Z, the last character of the file.

/A specified with the destination file puts an end-of-file character at the end of the file.

/A is the default when you concatenate files using COPY.

/B

specified with the source file copies the entire file to the physical end-of-file defined by the directory file size. For example:

```
copy b:new.fil /b cop.fil /a(cr)
```

In the preceding command, /B on the source file keeps the end-of-file mark (CTRL-Z), and /A on the destination file inserts a CTRL-Z. If you specify /B with the destination file, CTRL-Z is not placed at the end of the file.

By default, MS-DOS concatenates files as ASCII (text) files. A CTRL-Z (1A in hexadecimal) in the file is interpreted as an end-of-file mark.

When you want to combine binary files, use /B. /B forces the COPY command to use the physical end-of-file defined in the directory. /B remains in effect until MS-DOS encounters another /A or /B. For example, this command concatenates COP.FIL to the end of BIN.FIL and places the result into BIN.FIL as a binary file:

```
copy /b bin.fil + cop.fil(cr)
```

The COPY command is not a linker; object modules cannot be combined into an .EXE or a .COM file.

/V

is a command switch that tells MS-DOS to verify that the sectors written on the destination disk are recorded correctly. Use this switch to check that critical data is copied accurately.

When you use /V, COPY performs more slowly because extra time is needed to verify the COPY operation.

You can also use reserved device names (see Chapter 2.2.2) in a COPY command. For example, you can copy your keyboard input into a disk file with COPY CON:

```
A>copy con text.fil(cr)
This is a text file(cr)
Each time you press the Return key, a line is
created in the file(cr)
The last line ends with a CTRL-Z (and a Return)(cr)
^Z(cr)
A>_
```

After you press CTRL-Z (^Z) and Return, the COPY ends and the file is saved as TEXT.FIL.

Change Input/Output Devices (CAUX, CLST, CTTY)

CAUX [devicename](cr)

CLST [devicename](cr)

CTTY [devicename](cr)

CLST, CAUX, and CTTY are internal commands that define the devices (or device drivers) in your system. CAUX controls the auxiliary input/output device; CLST controls the list device (printer); CTTY controls the console device (keyboard and screen). If you enter one of these commands without specifying a device name, the current device driver is displayed.

devicename
: is the name of a device driver. You must use a valid device name for a device driver that was loaded in your MS-DOS or from a DEVICE command in your ICONFIG.SYS or VCONFIG.SYS file. MS-DOS installs the following device drivers:

  PRN is the list device for CLST
  AUX is the auxiliary I/O device for CAUX
  CON is the console I/O device for CTTY

To prevent confusion between device names and filenames, precede device names with \DEV\. For example, CTTY = CON is equivalent to CTTY = \DEV\CON.

To select an alternate device driver, type its device name with the appropriate command, such as:

```
clst fastprtr(cr)

caux cardrdr(cr)

ctty remote(cr)
```

You can change the port used for the list output by using one of these device names with CLST:

  AUX or COM1 to select serial port 1
  COM2 to select serial port 2
  LPT1, LST, or PRN to select the parallel port

For more information about your system's device drivers, see the DISKID file on your system diskette.

Current Date (DATE)

DATE [mm/dd/yy](cr) (in American operating systems)

or

DATE [dd-mm-yy](cr) (in European operating systems)

DATE is an internal command. MS-DOS records the date in the current directory for every file that you create or change. If you enter the DATE command without parameters, MS-DOS displays the day and the numeric representation for the month, the day, and the year last set. MS-DOS also displays the prompt for the new date:

```
Current date is Wed  6/22/85
Enter new date:_
```

If the date is incorrect, enter the new date. If you do not want to change the displayed date, press the Return key. You can also enter the DATE command with a date to change the date known to the system.

mm
: is the number for the month (1 through 12).

dd
: is the number for the day (1 through 31).

yy
: is the number for the year (80 through 99, or 1980 through 2099).

Separate the month, day, and year with hyphens (-) or slashes (/). If you use any symbols other than these, or if you use letters instead of numbers, the following message and prompt appear:

```
Invalid date
Enter new date:_
```

You can change the date from the keyboard or from a batch file. If you are using an AUTOEXEC.BAT file during MS-DOS startup that does not contain the DATE command, however, MS-DOS will not prompt you for the date. For this reason, you may want to include the DATE command in that file (see ICOMMAND/VCOMMAND in this chapter).

Delete Files (DEL)

DEL [path]filename.ext(cr)

DEL is an internal command that deletes all files that match the specified filename.

path
: is the path through the subdirectory system to the file you want to delete. Include the drive name in the path if the file is not on the default drive.

Enter DEL *.* to delete all the files. MS-DOS displays:

```
Are you sure (Y/N)?_
```

Type Y or N followed by a Return to confirm or retract your DEL *.* command.

The following command deletes the file EXAMP.TXT from the TEXT subdirectory on drive B:

```
del b:\bin\text\examp.txt(cr)
```

The following command deletes all the files with .LST extension from the current directory on the default drive:

```
del *.lst
```

Display Directory (DIR)

```
DIR [path][[filename][.ext]] [/P] [/W](cr)
```

DIR is an internal command. DIR without parameters lists all the files in the current (sub)directory on the default drive.

path
: is the path through the subdirectory system to the subdirectory you want to list, or to the file(s) whose existence you want to verify with DIR. The path includes the drive name and subdirectory names as necessary.

filename
: lists all the files with the given name.

filename.ext
: lists only those files from the root directory with the given name and extension.

/P
: stops listing the directory at one page or screenful of information. MS-DOS prompts you to strike any key to continue the listing.

/W

displays five filenames per line across the width of the screen. Only the filenames are displayed.

DIR lists the filenames, their sizes in bytes, and the time and date they were last modified, for all the parameters except /W.

You can use the ? and * wild-card characters (see Chapter 2.5) in the place of parts of a filename. For example:

dir *.*	is the same as	**dir**
dir filename	is the same as	**dir filename.***
dir .ext	is the same as	**dir *.ext**

The following example shows a sample directory listing. The backslash indicates that this is the root directory on drive A. When your current or default directory is the root directory on drive A, you can enter the DIR command without specifying a drive name.

```
A>dir(cr)

Volume in Drive A is JANUARY
Diskette is V format
Directory of A:\

STATUS     DAT       10368    1-23-85    2:12p
COST       DAT          31    1-31-85   11:06a
STATUS     BAK       10334    1-16-85    3:32p
CALENDAR   JAN        4068    1-12-85   12:30p
MEMOS           <DIR>         1-03-85   12:43p
LETTERS         <DIR>         1-10-85   10:04a

           6 File(s)   141516 bytes free

A>_
```

This example shows the same directory listing using the /W option:

```
A>dir/w(cr)
Volume ID for drive A: is JANUARY
Diskette is V format
Directory A:\
STATUS DAT   COST   DAT   STATUS BAK   CALENDAR DAT  MEMOS
LETTERS
       6 File(s)   141516 bytes free

A>_
```

Disk Comparison (DISKCOMP)

(I mode only)

DISKCOMP [d:] [d:] [/1] [/8](cr)

DISKCOMP is an external command that compares the contents of a diskette in one drive with the contents of a diskette in another drive. DISKCOMP can be used only with floppy diskettes. DISKCOMP always compares two entire diskettes to each other.

d:

is an optional drive letter giving the names of the diskettes that you want to compare.

/1

compares only the first side of two disks even if double-sided diskettes are being used.

/8

compares only 8 sectors per track even if 9-sector-per-track diskettes are in use.

Use the DISKCOMP command to verify that the DISKCOPY command has accurately copied the contents of one diskette onto another diskette.

You should not use DISKCOMP when the COPY command was used to copy a diskette. Although the contents of the two diskettes created with COPY might be identical, the arrangement of the data on each diskette might not be exactly the same. In this case, DISKCOMP indicates that corresponding tracks on the disks do not compare even though all files might have been accurately copied.

When you enter DISKCOMP, you can specify one or two drives. If you specify only one drive, DISKCOMP performs a single-drive comparison. Use this command if you have a system with one diskette drive and a fixed disk drive. MS-DOS prompts you to insert diskettes at the appropriate time. After giving the prompt, MS-DOS waits for you to press any key.

If DISKCOMP finds a discrepancy between the disks, you will see this message:

```
Compare Error(s) on
Track nn, Side x
```

where *nn* is the track number (0 through 39), and *x* is the side of the diskette (0 or 1) on which the discrepancy was found.

If the disks compare exactly, the following message is displayed:

```
Diskettes compare OK
```

If you do not specify any parameters, a single-drive comparison is done using the default drive. If the second drive name is omitted, the default drive becomes the second drive. If the default drive is specified as the first drive name, a single-drive procedure is used.

When you attempt to compare diskettes on a single-drive system, all prompts are for drive A, regardless of the parameters you specify.

Dual Drive Disk Copy (DISKCOPY)

(V mode only)

DISKCOPY [side to side](cr)

DISKCOPY is an external command that copies the contents of a VICTOR format diskette in one drive onto a diskette in another drive. You do not have to prepare the destination diskette with FORMAT before you use DISKCOPY; DISKCOPY formats the destination diskette automatically. During the copying process, DISKCOPY displays the number of the track being copied.

If you enter DISKCOPY without parameters, the program prompts you to:

- ▶ Name the drive (left or right) containing the diskette to be copied.
- ▶ Press the space bar to start the copy process.
- ▶ Repeat the process after the copy is complete.

The following example copies the diskette in the left drive (A) to the diskette in the right drive (B). Screen prompts are shown in the order they appear on the screen, but their position on the screen is not shown in the example.

```
A>diskcopy(cr)

Diskette COPY Utility -- Version ms x.y

Copy from FLOPPY drive? (Left or Right; press RETURN
to end)l

Copy from Left FLOPPY drive to Right FLOPPY drive.
Press space bar when ready.(sp)

Copy from Left FLOPPY drive to Right FLOPPY drive

Copy from Left FLOPPY drive to Right FLOPPY drive
complete.

Copy from FLOPPY drive? (Left or Right; press RETURN
to end)(cr)

A>_
```

If you follow the steps shown in the previous example, you can remove the system diskette from the left drive after DISKCOPY loads into memory. This also means that once the "Diskette COPY Utility" sign-on message appears, you can use either drive for the source or destination diskette. In this way you can load DISKCOPY from your system diskette and then insert another diskette that you want to copy.

side to side

is an optional phrase naming the drives containing the source and destination diskettes. This phrase pertains to two-drive systems. If you have a one-drive system with a fixed disk, insert the destination diskette into the drive and enter the appropriate drive letter. The phrase can take either of these two forms:

left to right

right to left

You can abbreviate this phrase to letters. You can use L and R to represent left and right, and you can use T instead of TO, or you can omit the T entirely. For example, these three commands copy the diskette in the right drive to the diskette in the left:

```
diskcopy r to l

diskcopy r t l

diskcopy r l
```

If you include the drive sides in a DISKCOPY command, DISKCOPY loads into memory and begins copying immediately, so that you have no opportunity to swap diskettes. For this reason, you should include the drive parameters only if you are copying your system diskette (the diskette containing DISKCOPY).

If there have been many file deletions and creations on the diskette you are copying, the disk is fragmented. Fragmentation occurs because disk space is not always allocated sequentially when files are created. Access time for a fragmented diskette is slower than for a diskette with consolidated file. You can consolidate files on a fragmented diskette by using the COPY command instead of DISKCOPY.

For example, the following COPY command copies all the files from the diskette in drive A to the diskette in drive B:

```
copy a:*.* b:(cr)
```

Note that you cannot copy your operating system diskette using *.* because the system files are not copied.

Echo Messages to Display (ECHO)

ECHO ON(cr)

or

ECHO OFF(cr)

or

ECHO message(cr)

ECHO is an internal command for batch files that controls the display of batch file commands and comments during batch file processing. ECHO without parameters specifies the current setting (ON or OFF).

ON

displays all batch commands and remarks (REM) during batch file processing. ECHO ON is the default.

OFF

turns off display of the commands in the batch file. The command (such as A>ECHO OFF) is displayed, but any batch commands following this command are not displayed.

message

is a string of alphanumeric characters that you want displayed during batch file processing, for example:

```
copy con greet.bat(cr)
echo Welcome aboard!(cr)
echo Enter your name.(cr)
^Z(cr)
```

These commands copy the two ECHO command lines to a batch file called GREET.BAT. A CTRL-Z (^Z) closes the file and returns you to the MS-DOS system prompt.

When you specify GREET, MS-DOS displays:

```
A>echo Welcome aboard!
Welcome aboard!
A>echo Enter your name.
Enter your name.
```

You can include ECHO OFF in a batch file to suppress just the display of the batch commands. And, in the same file, you can specify ECHO with a message to display that message during batch processing. For example, if you add ECHO OFF to the GREET.BAT file above:

```
copy con greet.bat(cr)
echo off(cr)
echo Welcome aboard!(cr)
echo Enter your name.(cr)
^Z(cr)
```

and enter GREET from the keyboard, MS-DOS displays only the messages following each ECHO command:

```
A>echo off
Welcome aboard!
Enter your name.
```

Erase Files (ERASE)

ERASE [path]filename.ext(cr)

ERASE is an internal command that deletes all files that match the specified filename. See the DEL command in this chapter.

Convert Executable Files to Binary Files (EXE2BIN)

EXE2BIN [path]filename.ext [[path][filename[.ext]]](cr)

EXE2BIN is an external command that converts linker-generated .EXE (executable) files to binary (.COM) format. This procedure saves disk space and provides faster program loading.

The resident (actual code and data) part of the .EXE file must be less than 64K. The input file is converted to .COM file format (memory image of the program) and placed in the output file. The input file must be in valid .EXE format produced by the linker. There must be no STACK segment.

path
  is the path through the directory hierarchy to the file that you want to specify as the input file or the output file.

filename.ext
  is the name of the input file. It can also be the name of the converted file. If you do not specify a file extension with the input filename, it defaults to .EXE. If you do not specify an output filename, the input filename is used. If you do not specify a file extension in the output filename, the new file's extension is .BIN.

Two kinds of conversions are possible, depending whether the initial CS:IP (Code Segment:Instruction Pointer) is specified in the .EXE file:

1. If CS:IP is not specified in the .EXE file, a pure binary conversion is assumed. If segment fixups are necessary (for example, the program contains instructions requiring segment relocation), you will be prompted for the fixup value. This value is the absolute segment at which the program is to be loaded. You can use the resulting program only when loaded at the absolute memory address specified by a user application. The command processor is not capable of loading the program correctly.

2. If CS:IP is specified as 0000:100H, it is assumed that the file is to be run as a .COM file with the location pointer set at 100H by the assembler statement ORG; the first 100H bytes of the file are deleted. No segment fixups are allowed because .COM files must be segment relocatable.

Once the conversion is complete, you can rename the resulting file with a .COM extension. Then the command processor can load and execute the program in the same way as the .COM programs supplied on your MS-DOS diskette.

Change EXE Header Size (EXESIZE)

```
EXESIZE filename [/Nxxxx] [/Xyyyy] [/P](cr)
```

The EXESIZE command is an external command that sets a specified file's EXE header values so that it can use MS-DOS memory allocation functions.

If your program needs to allocate additional memory space, or it requires statically allocated memory, specify the hexadecimal number of 16-byte paragraphs with the /N and /X options. If your program's linker-produced EXE header value does not explicitly define statically allocated memory, such as the stack/heap area of some high-level languages, use /N and /X.

If you do not specify /N or /X, EXESIZE assumes that the file does not need additional memory space, and thus sets the memory space required to the size of the EXE file itself.

filename
: is the name of the file for which you want to set the EXE header values. If you do not specify a file extension, it is assumed that the file is an .EXE file.

/N
  is the minimum amount of memory space needed for the program to run, where *xxxx* is any hexadecimal number.

/X
  is the maximum amount of memory space that the program can use to run, where *yyyy* is any hexadecimal number.

/P
  causes the screen display to pause if EXESIZE encounters an error.

Exit ICOMMAND.COM/VCOMMAND.COM (EXIT)

```
EXIT(cr)
```

EXIT is an internal command. You can use this command to return from ICOMMAND.COM/VCOMMAND.COM to the program or to the command processor you were using previously.

If you are in V mode, and MS-DOS is processing a program that allows you to call the VCOMMAND.COM program, you enter:

```
vcommand(cr)
```

to invoke VCOMMAND.COM. You can now use any internal command. To return to the previous program, specify:

```
exit(cr)
```

Note: If you are in I mode, you specify ICOMMAND to call ICOMMAND.COM, but the EXIT command remains the same.

Fast Expression Search (FGREP)

FGREP [options] expression [files or directories](cr)

FGREP.EXE is an external command (utility) that searches through a list of files for files that contain a specific text string. FGREP outputs the lines that contain the text string and the names of the files. Options allow you to search subdirectories, list only filenames, and list the number of lines that match.

Options can be any of the following:

/^
: matches only text at the beginning of a line.

/$
: matches only text at the end of a line.

/C
: prints only a count of the lines containing matching text.

/I
: ignores upper/lowercase when matching the expression.

/L
: also prints line numbers for matching text in files.

/N
: prints only the filenames of files with matching text.

/R
: recursively searches subdirectories of the directories in the [files or directories] list. If you do not include this option, FGREP searches only the files you name in the command, as well as all files in any directories named in that command.

/V

reverses the match—prints all lines except those with a matching expression.

/W

matches only text that is a word (or two words) not preceded by a letter, digit, or underscore.

expression

is the text string to search for. If the expression does not contain spaces or wild-cards, you can specify this string exactly as you want it to appear in the files—that is, without delimiters such as quotes. For example:

```
fgrep jack a:chapt1.doc(cr)
```

searches for "jack" in the file CHAPT1.DOC on drive A.

If the expression contains spaces, hyphens, or MS-DOS command-line characters (such as < or >), you must enclose the string in double quotes. For example:

```
fgrep "a jack-in-the-box" *.pas(cr)
```

searches all files with the .PAS extension for "a jack-in-the-box", an expression containing spaces and hyphens.

files or directories

is a list of one or more subdirectory names and/or filenames which can include wild-card characters. This list specifies the domain on which the FGREP command will operate. If no files are given, FGREP accepts keyboard input or redirected input to search.

For example:

```
fgrep /i^$ proc c:\sources d:\project\*.asm(cr)
```

ignores upper/lowercase (/I) and prints a list of files and their lines that begin (/^) and end (/$) with (i.e., consist only of) the string "proc".

Files in the directory C:\SOURCES and in the D:\PROJECT directory with the .ASM file extension are searched.

This command:

```
fgrep /ri proc c:\sources d:\project\*.asm(cr)
```

is similar to the previous example, except that any subdirectories (/R) of C:\SOURCES and their subdirectories will also be searched.

The following command:

```
fgrep /vn priority c:\memos d:\*.not(cr)
```

outputs the names of files (/N) which have lines that do not contain (/V) the word "priority". FGREP searches the \MEMOS subdirectory on drive C, and all files with .NOT extension in the root directory on drive D.

File Comparison (FILCOM)

```
FILCOM [/#] [/B] [/C] [/W] file1 file2(cr)
```

File Comparison (FILCOM.EXE) compares the contents of two files. You can abbreviate FILCOM to FC. You can compare disk files if you have two copies and you want to see which one is current. FILCOM outputs the differences between the two files to the console or to a third file. The files can be source files (files containing source statements of a programming language) or binary files (output by an assembler, the linker, or a high-level language compiler).

The comparison can be made in one of two ways: line by line or byte by byte. The line-by-line comparison isolates blocks of lines that are different between the two files and prints those blocks of lines. The byte-by-byte comparison displays the bytes that are different between the two files.

FILCOM matches FILE1 against FILE2 and reports any differences between them. Both filenames can include pathnames. For example:

```
fc b:\mgr1.nam\ted\f1.asm  \mgr2.nam\f2.asm(cr)
```

compares F1.ASM in the \MGR1.NAM\TED subdirectory on drive B with F2.ASM in the \MGR2.NAM subdirectory on the disk in the default drive.

/#

is a number from 1 to 9. This switch specifies the number of lines required to match for the files to be considered as matching again after a difference has been found. If /# is not specified, it defaults to 3. /# is used only in source file comparisons.

/B

forces a binary comparison of both files. The two files are compared byte-to-byte, with no attempt to resynchronize after a mismatch. The mismatches are printed as follows:

```
--ADDRS----F1----F2--
xxxxxxxx    yy    zz
```

where *xxxxxxxx* is the relative address of the pair of bytes from the beginning of the file. Addresses start at 00000000; *yy* and *zz* are the mismatched bytes from FILE1 and FILE2, respectively. If one of the files contains less data than the other, a message is printed out. For example, if FILE1 ends before FILE2, FILCOM displays:

```
***Data left in F2***
```

/C

ignores the case of letters. All letters in the files are considered uppercase. This switch is used only in source comparisons.

For example:

Much MORE data IS NOT FOUND

matches with:

much more data is not found

/W

compresses "whites" (tabs and spaces) during the comparison. Thus, multiple contiguous spaces in any line are considered a single space. Note that although FILCOM compresses spaces, it does not ignore them. FILCOM ignores beginning and ending spaces in a line. /W is used only in source comparisons. In this example an underscore represents a blank space:

____More___data_to_be_found_____

matches with:

More_data_to_be_found

and with:

_______More_____________data_to_be_______found________

but will not match with:

Moredata_to_be_found

If both /W and /C are specified, FILCOM compresses tabs and spaces, and ignores case. For example:

____DATA_was_found____

matches with:

data_was_found

file1 file2

are the full filenames and paths for the two files you want to compare. If the file is in the current directory on the default drive, you can omit the drive and directory names for that file.

FILCOM reports the differences between the two files you specify by displaying the first filename, followed by the lines that differ between the files, followed by the first line to match in both files. FILCOM then displays the name of the second file followed by the lines that are different, followed by the first line that matches. For example:

```
          ...
          ...
----------<filename1>
<difference>
<1st line to match file2 in file1>

----------<filename2>
<difference>
<1st line to match file1 in file2>
------------------------------------------

          ...
          ...
```

FILCOM continues to list each difference.

If there are too many differences (involving too many lines), the program reports that the files are different and stops.

If there are no matches after the first difference is found, FILCOM displays:

```
*** Files are different ***
A>_
```

The differences and matches between the two files are displayed on your screen unless you redirect the output to a file. You can redirect FILCOM output with the > and < symbols as described in Chapter 5. For example, to compare F1.TXT and F2.TXT and then send the FILCOM output to DIFFER.TXT, type:

fc f1.txt f2.txt > differ.txt(cr)

FILCOM uses a large amount of memory as buffer (storage) space to hold the source files. If the source files are larger than available memory, FILCOM compares what can be loaded into the buffer space. If no lines match in the portions of the files in the buffer space, FILCOM displays only the message:

```
*** Files are different ***
```

For binary files larger than available memory, FILCOM compares both files completely, overlaying the portion in memory with the next portion from disk. All differences are output in the same manner as those files that fit completely in memory.

Find Alphanumeric Characters (FIND)

```
FIND [/C] [/N] [/V] "string" [path]filename.ext...(cr)
```

FIND.EXE is an external command for filtering data. This command searches a file to find a specified string of alphanumeric characters and displays all the lines on the screen that contain the specified string.

/C
: displays only the number of matches with the string. The lines that match the string are not displayed.

/N
: displays the relative number of the line(s) that match the string. FIND displays the relative line number, then the line.

/V
: displays the lines that do not contain the string.

"string"
: is a string of alphanumeric characters that you want to find. The characters must be enclosed in quotes. If they are already enclosed in quotes, use double quotes to specify the characters as the string parameter.

path
: is the path through the directory hierarchy to the file(s) to be searched for the string.

filename.ext
: is the name of the file or files to be searched for the specified string.

The following example searches a mail list file for "Washington, D.C." You can create MAIL.LST by entering:

```
copy con mail.lst(cr)
George Washington, Washington, D.C.(cr)
Charles de Gaulle, Paris, France(cr)
Pierre Trudeau, Montreal, Canada(cr)
Abraham Lincoln, Washington, D.C.(cr)
Margaret Thatcher, London, England(cr)
Ronald Reagan, Washington, D.C.(cr)
^Z(cr)
```

Then enter:

```
find "Washington, D.C." mail.lst(cr)
```

On the first line MS-DOS displays a dashed line, the path to the file if one was specified, and the name of the file:

```
----------mail.lst
```

MS-DOS then displays each line containing the string "Washington, D.C."

If MS-DOS cannot locate the FIND command, MS-DOS displays:

```
Bad command or filename
```

FIND is an external command that may be in another subdirectory. Enter a PATH command giving the location of the external command FIND.

If MS-DOS displays:

```
FIND: File not found mail.lst
```

the MAIL.LST file is in a subdirectory. You need to specify the path to the file.

The FIND command can also filter data from an input source and pipe it to an output source (see Chapter 5.3.1). For example, you can find the characters "Washington, D.C." and pipe them to an output file called USPRES.DAT:

find "Washington, D.C." mail.lst | sort > uspres.dat(cr)

The | tells MS-DOS to use the output from the FIND command as input to the SORT command.

Use the DIR command after you have piped data to verify that the file is in the directory that you want. If it is not, use the COPY command to put the file in the correct directory.

For Each One Repeat MS-DOS Command (FOR–IN–DO)

FOR %%variable IN (set) DO command %%variable(cr)

FOR is an internal command used in batch files. It can also be used interactively. FOR repeats the specified MS-DOS command for each member in the specified set.

%%variable
: is a dummy variable where "variable" can be any character except the numbers 0–9 (which are used for the replaceable batch parameters, %0–%9). %%variable is replaced by each sequential member in the specified set. The variable specified at the beginning of the command line must be the same as the last variable in the command line. For example, if you specify %%A for the first variable parameter and %%F for the last variable parameter, MS-DOS displays:

```
Entry error
```

IN
: tells MS-DOS to sequentially replace %%variable with each sequential member in the specified set.

(set)
: can be any alphanumeric characters, including the replaceable batch parameters %0–%9. The set contains more than one member and must be enclosed in parentheses:

(mgr1 mgr2 mgr3)

You cannot specify a path in the set.

DO
tells MS-DOS to process the command that follows.

command
is an MS-DOS command.

For example, with FOR you can create a batch file that makes a new directory, changes the directory, and makes subdirectories:

```
copy con mgrdir.bat(cr)
mkdir %1(cr)
chdir %1(cr)
for %%a in (%2 %3 %4) do mkdir %%a(cr)
^Z(cr)
```

To process the file, enter the batch filename (MGRDIR) and the subdirectory names you want to create:

```
mgrdir managers mgr1 mgr2 mgr3(cr)
```

MANAGERS replaces %1, MGR1 replaces %2, MGR2 replaces %3, and MGR3 replaces %4. %%A is replaced by the first member of the set, which is MGR1. The command then tells MS-DOS, "FOR MGR1 IN (MGR1 MGR2 MGR3) DO MKDIR MGR1." MGR1 (%%A) is then replaced by MGR2, and MGR2 is replaced by MGR3.

MS-DOS displays each batch command as it executes:

```
A>mkdir managers
A>chdir managers
A>for %a in (mgr1 mgr2 mgr3) do mkdir %a
A>mkdir mgr1
A>mkdir mgr2
A>mkdir mgr3
A>_
```

Format Diskette (FORMAT)

(V mode only)

```
FORMAT [side:] [/D](cr)
```

FORMAT is an external command that prepares floppy diskettes in VICTOR format to receive data. As it formats the storage space on a diskette, FORMAT erases any previous files. You can load the FORMAT program, like the DISKCOPY program, in either of two ways—by typing only the program name, or by typing a full command line. Like DISKCOPY, FORMAT has a full-screen mode that enables you to process more than one diskette without reloading the program for each one.

If you enter:

```
format(cr)
```

the program loads into memory and displays a screen that asks you to:

- ▶ Select the drive (left or right) containing the diskette to be formatted.
- ▶ Specify an optional Volume ID (an on-disk label of up to 11 alphanumeric characters) to identify the diskette.
- ▶ Select double-sided format if your computer has double-sided drives.

To make selections, use the cursor keys on the right side of the keyboard. Then use the function keys at the top of your keyboard to start or quit the FORMAT process. The bottom line of the screen shows the current value of the function keys. For example, you can get onscreen HELP by pressing F7.

If you include the drive side in your FORMAT command line, the program formats the diskette on the side you name and then exits to the operating system. For example, you can type:

```
format /d right(cr)
```

The FORMAT program displays its one-line sign-on message and immediately begins formatting the diskette in the right drive (drive B). FORMAT tells you when the process is complete, and then redisplays the system prompt. The drive side can be abbreviated to L (left) or R (right).

The /D switch with the FORMAT command formats a double-sided diskette. If you do not specify /D, your diskette is formatted as single-sided.

Go To Label (GOTO)

GOTO label(cr)

GOTO is an internal command for batch files. Batch processing continues with the command on the line below the specified label. GOTO is issued as a separate command except when used with the IF command. IF uses GOTO on the same command line as IF.

label
: is a string of up to 8 alphanumeric characters. The label in the GOTO command must match a label in the batch file. The label in the batch file must be preceded by a colon, telling MS-DOS to ignore this line.

  The line containing the label can precede or follow the GOTO command in the batch file. When the label precedes GOTO, the batch file goes into a loop. When the label follows GOTO, batch processing skips the intervening commands. If the batch file contains a GOTO command that does not include a label, the batch file ends.

In the following example, the double = tells MS-DOS that this is an "equal to" symbol, not an assignment command, such as SET PATH = A:\COMDIR.

```
copy con private.bat(cr)
echo off(cr)
if %1 == security goto greet(cr)
echo Access denied(cr)
goto end(cr)
:greet(cr)
echo Welcome! Enter your name!(cr)
:end(cr)
^Z(cr)
```

To process PRIVATE.BAT and to substitute the word SECURITY for the replaceable parameter %1, type:

```
private security(cr)
```

MS-DOS displays:

```
A>echo off
Welcome! Enter your name!
```

If you specify any word other than SECURITY, MS-DOS displays:

```
A>echo off
Access denied
```

The ECHO OFF command stops the display of the batch commands during batch file processing. MS-DOS replaces %1 with the name you entered when you specified the batch file. MS-DOS compares the name to the one in the IF command. When there is a match, the welcome greeting and the prompt message for a name are displayed. If there is no match, the next command processed is the ACCESS DENIED message. MS-DOS then goes to the command following the label :END, which is the end-of-file marker (^Z).

Note: The labels are not displayed during batch file processing because MS-DOS ignores lines that start with a colon. You can add non-displayed comments to a batch file by starting each comment line with a colon.

Print Graphics Display Screen (GRAPHICS)

(I mode only)

```
GRAPHICS(cr)
```

GRAPHICS is an external command that allows the contents of a graphics display screen to be printed on printers that support the printing of individual dots.

Enter the GRAPHICS command after loading MS-DOS. Any time you want a printout of the screen, press SHIFT-PRT SC.

If the screen is in text mode, the printout will be in text mode.

If the screen is in the 320 × 200 color graphics mode, each screen dot is printed in one of four shades of gray.

If the screen is in the 640 × 200 color graphics mode, the screen is printed sideways on the paper.

Printing speed is dependent on the speed of the printer being used.

Display Command History (HISTORY)

HISTORY [n](cr)

HISTORY is an internal command that captures console input. HISTORY remembers the commands you have entered. With HISTORY you can display the command history, and you can select any command from the history to re-enter into the MS-DOS command-line template. Then, using the function keys as described in Chapter 5, you can edit and/or re-enter that command without retyping the entire line.

MS-DOS can store a history of any number of command lines, up to 256 characters. Typically, HISTORY retains from 10 to 25 commands. HISTORY commands themselves, however, are not stored in the command history. You can abbreviate HISTORY to HI.

Entering HISTORY with no parameters displays the command history onscreen, like this:

```
n.   <oldest line in HISTORY>
.       .      .      .
.       .      .      .
2.   <command entered before #1>
1.   <command entered before this HISTORY command>
```

Commands too long to fit the display line wrap to the next line.

n

is the number of a command stored in the history. The nth command from the history is placed into the command-line template, where you can edit and/or execute it as shown in Chapter 5.

For example, entering:

```
hi 5(cr)
```

puts the fifth line from the command history into the command-line template. If you want to execute that command without changing it, press F3 (Copy Template) in I mode or F7 (Copy Template) in V mode.

HISTORY 1 refers to the last command entered before any HISTORY command(s). HISTORY 2 is always the non-HISTORY command before that.

Reload Command Processor (ICOMMAND/VCOMMAND)

```
ICOMMAND [path][device] [/D] [/P] [/C string](cr)
VCOMMAND [path][device] [/D] [/P] [/C string](cr)
```

ICOMMAND/VCOMMAND is an external command that invokes the command processor, ICOMMAND.COM or VCOMMAND.COM depending on the mode you are using: I mode or V mode. When you enter ICOMMAND or VCOMMAND, MS-DOS reloads the appropriate command processor.

You can use ICOMMAND or VCOMMAND interactively by entering the command from the keyboard at the system prompt, or you can use ICOMMAND or VCOMMAND in batch files to reload the command processor.

path

is the path through the directory system to the file containing the command processor. The path includes the drive name and subdirectory names where necessary. No path is necessary if the file resides in the root directory on the default drive.

device

is the name of the TTY device. If you do not specify a device, MS-DOS uses CON (by default, the keyboard and screen).

/D

omits prompts for date and time when the command processor reloads. If you do not specify /D, you are prompted for the date and time.

/P

places ICOMMAND.COM/VCOMMAND.COM in memory until you reload the operating system. /P also tells the command processor to search for an AUTOEXEC.BAT file and to prompt you for the date and time as it loads into memory. If /P is omitted, ICOMMAND.COM or VCOMMAND.COM is a transient command; you can exit the current command processor loaded in memory and return to the previously loaded command processor.

/C string

submits a string for the command processor to execute after it loads into memory. ICOMMAND.COM/VCOMMAND.COM executes this single command string and then exits. The /C switch overrules the /P switch. If present, the /C switch must be the last switch. All text following /C is passed to the command processor.

Intentionally
blank

When you specify USERS and a name to be processed, MS-DOS replaces the %1 parameter in the first IF command with the specified name. MS-DOS compares that name (string1) to the string of characters in string2. If they are equal, MS-DOS displays:

```
Welcome! Enter your command.
```

If there is not a match in the first command, the next command is processed. If MS-DOS processes the three IF commands in the file without a match, MS-DOS displays:

```
You must be an approved user to operate the system.
```

EXIST [path]filename.ext

tells MS-DOS to look for the specified file in the specified path. If you do not include a path, MS-DOS searches the current directory on the default drive. If the file is found, the command is processed. For example, this command in a batch file:

```
if not exist \mgr\jobs echo Can't find \mgr\jobs.(cr)
```

searches for the file JOBS in the MGR subdirectory on the disk in the default drive. If JOBS is found, MS-DOS does not process the ECHO command. If \MGR\JOBS does not exist, MS-DOS displays:

```
Can't find \mgr\jobs.
```

ERRORLEVEL n
tests for a program failure. This parameter tells MS-DOS to check for an error level number, where *n* is the specified number. The next command is processed if the previous program run by MS-DOS had an exit code the same as *n* or higher.

command
can be any of the MS-DOS commands. The most common command used with IF is GOTO.

List Files (LS)

LS [options] [files or directories](cr)

LS.EXE is an external command that produces a sorted listing in various directory formats of a file or list of files. In the LS listing, names of subdirectories are followed by a \ (such as MGR1\).

Options can be any of the following switches:

/B
sorts backwards (Z to A).

/E
sorts output by file extension.

/G
lists files "toGether." Files are not separated by directory.

/L
lists the directory entries in long format. The filename, file attribute, date, time, and size are listed for each file. Long-format listings are not sorted.

/R

recursively lists all files in subdirectories of any directory specified. You can use this switch to list the contents of all directories on a drive. LS /R lists each subdirectory followed by all the files in that directory.

/T

sorts output by date and time of creation, rather than alphabetically.

/1

lists only one filename per line.

files or directories

is a list of one or more filenames and/or directories. If this parameter is omitted, LS lists the current directory.

The following example alphabetically lists the files and subdirectories in the current directory:

ls(cr)

The next example lists, in long format, the files in the \TEMP directory (and its subdirectories) on drive B, and any file with .ASM extension in the working directory:

ls /rl b:\temp *.asm(cr)

The listing on your screen will be similar to the following:

```
\temp:
a----w    1430   24-SEP-84   12:57 BLDTYPE.ASM
a----w     616   20-MAR-85   10:48 BRITE.ASM
a----w    6247   02-FEB-85   12:37 DTIME.ASM
a----w   60491   10-OCT-84   10:48 HMANAGER.ASM
a----w     229   09-FEB-85   17:44 OMANAGER.ASM
5 files.
```

Make Directory (MKDIR)

MKDIR [path](cr)

MKDIR is an internal command that makes a new subdirectory beneath the root directory, which MS-DOS created when you formatted your disk. You can abbreviate MKDIR to MD. See Chapter 3 for more information on creating and using subdirectories.

path
: is the path through the subdirectory hierarchy to the subdirectory you want to create. The last directory name in the path is the name of the new subdirectory. MS-DOS can create a subdirectory only if a directory above it exists. An initial \ always indicates the root directory. An initial directory name without a \ indicates a subdirectory below the current directory.

For example, if MS-DOS is at the root directory and you enter:

```
mkdir mgr1.nam(cr)
mkdir mgr2.nam(cr)
mkdir mgr3.nam(cr)
```

MS-DOS creates these three subdirectories at the first level beneath the root directory. Each subdirectory is listed in the root directory. You can verify this by entering the DIR command. MS-DOS displays:

```
Volume in drive A is June 8 1985
Directory of A:\
MGR1      NAM <DIR>     6-30-85   9:00a
MGR2      NAM <DIR>     6-30-85   9:00a
MGR3      NAM <DIR>     6-30-85   9:01a
        3 File(s)   370683 bytes free
```

"Directory of A:\" indicates the root directory. Note that subdirectories are displayed with <DIR>; files are not displayed with this attribute.

Once a directory exists, you can make a subdirectory below it. For example, this command creates the subdirectory JOBS below the subdirectory MGR1.NAM:

```
mkdir mgr1.nam\jobs(cr)
```

You can enter a directory name without a \ in a MKDIR command to add a subdirectory to the directory you are currently using. For example, if you are working in the MGR1.NAM directory, instead of issuing the preceding command, you can specify:

```
mkdir jobs(cr)
```

Once the JOBS subdirectory has been created, you can create a subdirectory below it:

```
mkdir mgr1.nam\jobs\benefits(cr)
```

or, if the current directory is JOBS, enter:

```
mkdir benefits(cr)
```

You can make as many subdirectories as you want depending on the amount of space on your disk. Subdirectory names can be 1 to 8 characters followed by an extension of 1 to 3 characters. The valid characters for filenames are valid for directory names. You can duplicate file or directory names as long as they are in separate directories. When you are creating subdirectories, do not use more than 63 characters in the command, including the \.

When you finish making your directory hierarchy, it is a good idea to enter the CHDIR \ command. This command returns you to the root directory. You should always return to the root directory so that you do not accidentally create an entry in the wrong subdirectory.

Modify Console (MODCON)
(V mode only)

MODCON sourcefile [savefile](cr)

MODCON.EXE is an external command for modifying the "soft" keyboard and character set used by the console. A soft keyboard is a part of the operating system that tells the computer what character and code to generate for each keystroke. With MODCON you can select a new keyboard file and/or character set file. You can also save the current character set(s), and then restore them later.

sourcefile

is the file or files containing the sets to be made active. Follow these rules for the source filename:

▶ If the source file is in a subdirectory, include the path through the directory system to the file (see Chapter 3).

▶ To select only a keyboard file or only a character set file, include the .KB or .CHR extension for the source file.

▶ If the source filename has no extension, MODCON activates both the .KB and the .CHR files with that filename.

▶ If you enter an asterisk (*), no new sets are made active and the configuration remains unchanged.

savefile

is the file or files used to save the current keyboard table and/or character set. Follow these rules for the save filename:

▶ If you want to save the existing file(s) in a subdirectory, include the path through the directory system to the file or files.

▶ If you are saving only a keyboard file or only a character set file, the saved file should have .KB or .CHR extension, respectively.

▶ To save both keyboard and character set files, omit the extension for the saved filename.

Note: The extension for the source file is independent of the extension for the saved file.

Here are some example command lines:

```
modcon b:\dir3\test.chr \chr\save.chr(cr)
```

This command modifies the character set, but does not affect the keyboard. The new character set is TEST.CHR in the \DIR3 subdirectory on drive B. The current character set is saved in SAVE.CHR in the \CHR subdirectory on the default drive.

```
modcon \chr\test.kb(cr)
```

This command selects a new keyboard file, called TEST.KB in the \CHR subdirectory on the default drive, but does not save the current keyboard file.

```
modcon germ01 g02save(cr)
```

This command selects GERM01.KB and GERM01.CHR as the new keyboard and character set. The current keyboard table and character set are saved in G02SAVE.KB and G02SAVE.CHR on the default drive.

```
modcon germ01 germ01(cr)
```

This command activates the GERM01.KB and GERM01.CHR files on the default drive. The save operation overwrites the original GERM01.KB and GERM01.CHR files with the current character set and keyboard.

```
modcon aust01(cr)
```

This command makes AUST01.KB and AUST01.CHR the active keyboard table and character set. The original keyboard table and character set are not saved.

```
modcon * brit01.kb(cr)
```

This command saves the active keyboard set in the file BRIT01.KB on the default drive, but leaves it as the active set. The character set is not affected.

```
modcon french.kb b:swedish.chr(cr)
```

This command saves the current character set in B:SWEDISH.CHR and leaves that character set active. It also uses the keyboard from FRENCH.KB on the default drive, overwriting the current keyboard table in memory without saving it.

Character Sets and Keyboard Files

To use MODCON, you must have on disk the keyboard or character set files that you want to use in your configuration.

Character sets are obtained by:

- ▶ Selecting a set provided on the system selection (SYSGEN) diskette included in the *Applications Programmer's Tool Kit II*.
- ▶ Selecting a graphics character set provided with CHARGRAF in the *Graphics Tool Kit II*.
- ▶ Modifying the current set or any available set using the EFONT character-font editor included in the *Graphics Tool Kit II* and in the *Applications Programmer's Tool Kit II*.

Keyboard tables are obtained by:

- ▶ Selecting a table provided on the system selection (SYSGEN) diskette included in the *Applications Programmer's Tool Kit II*.
- ▶ Modifying the current table or any available table using the keyboard table editor KEYGEN included in the *Graphics Tool Kit II* and in the *Applications Programmer's Tool Kit II*.

Note: You can purchase the preceding tool kit packages from your dealer.

Set Operation Modes (MODE)

MODE is an external command that sets operation modes for the video display and for printer or serial ports. There are four options for this command. Each has its own function.

MODE LPT #:[n][,[m][,P]](cr)

Sets mode parameters for a printer

MODE n(cr)

Sets mode parameters for a video display, for I mode only

MODE COMn:baud[,parity[,databits[,stopbits[,P]]]](cr)

Sets mode parameters for a serial port

MODE LPT #: = COMn(cr)

Redirects parallel printer output to a serial port

1. Set Mode Parameters for a Printer

With this option you can set the operation mode for a printer. The syntax is:

MODE LPT#:[n][,[m][,P]](cr)

#	Establishes the printer number (1, 2, or 3).
n	Establishes the number of characters per line (80 or 132).
m	Establishes the number of vertical lines per inch (6 or 8).
[,P]	Establishes continuous retry on timeout errors.

If you do not specify a value for m or n, or if the value you specify is invalid, then the mode for that operation does not change. You can stop continuous retries by pressing CTRL-BREAK. If you do not want timeout errors to continuously retry, do not specify the P parameter.

For example, if you want to set line printer number 2 for continuous retry and 132 columns with 8 line per inch spacing, enter the following command:

```
mode lpt2:132,8,p(cr)
```

2. Set Mode Parameters for a Video Display (I mode only)

Option 2 sets the operation mode of the video display(s). The syntax is:

```
MODE n(cr)
```

n Establishes the type of display (40, 80, BW40, BW80, CO40, CO80, or MONO). If the color dipswitch is set, specifying MONO does nothing to the display. You must physically set the dip switch to change the type of display from monochrome to color.

40 =	40 character display width (color/graphics display mode)
80 =	80 character display width (color/graphics display mode)
BW40 =	Black and white, 40 character display width
BW80 =	Black and white, 80 character display width
CO40 =	Color, 40 character display width
CO80 =	Color, 80 character display width
CO320 =	Color/Graphics, 320 × 200 display
BW320 =	Color/Graphics, 320 × 200 display
BW640 =	Color/Graphics, 640 × 200 display
MONO =	Monochrome, 80 character display width

3. Set Mode Parameters for Serial Communications Ports

This option allows you to set the operation mode for serial ports. The syntax is:

```
MODE COMn:baud[,parity[,databits[,stopbits[,P]]]](cr)
```

n	Establishes the serial port number (1 or 2).
baud	Establishes the baud rate (110, 150, 300, 600, 1200, 2400, 4800, or 9600). Only the first two digits are required (for example, 11, 15, 30, 60, and so on).
parity	Establishes the parity mode (N = none, O = odd, E = even). The default is E (even).
databits	Establishes the number of bits in a serial byte (7 or 8). The default is 7.
stopbits	Establishes the number of bits signaling the end of a serial byte (1 or 2). If the baud rate is 110, the default is 2; otherwise, it is 1.
[,P]	This parameter causes continuous retries to occur on timeouts. This is useful when you use a serial printer that cannot keep up with the serial rate. Press CTRL-BREAK to stop the retry loop. If you do not want the timeout errors to continuously retry, reinitialize the serial port with the P parameter.

You must specify values for the serial port number and the baud rate; if you omit values for the other parameters, MODE assigns the default values. When you do not enter a parameter, enter a comma for the value omitted.

For example, to set up serial port number 2 for 9600 baud, even parity, 8 databits, 2 stopbits, and continuous retries, enter the following command:

```
mode com2:96,e,8,2,p(cr)
```

To use the default values for this command, enter the following:

```
mode com1:12,,,,p(cr)
```

4. Redirect Parallel Printer Output to a Serial Port

This option redirects output for a parallel printer to an asynchronous communications adapter. With this option, you can redirect output for the logical printers to the serial I/O ports. Before you use this option, you must use Option 3 to initialize the serial port. The syntax for Option 4 is:

```
MODE LPT#: = COMn(cr)
```

#	Establishes the printer number (1, 2, or 3).
n	Establishes the serial port number.

The following command causes all outputs to line printer number 3 to be redirected to serial port number 2:

```
mode lpt3: = com2(cr)
```

To disable the redirection for the printer number 3, enter:

```
mode lpt3:(cr)
```

More Screen (MORE)

MORE(cr)

MORE.COM is an external filter command that passes through the data in a file and displays one screen of data. At the bottom of the screen, this message is displayed:

```
-- More --
```

Press Return or the space bar to display another screen of data. The MORE message displays with each new screenful of data until MS-DOS reads the entire file.

MORE is used mostly with other commands to pipe data (see Chapter 5.3). For example, you can display a directory one screen at a time:

dir | more(cr)

The following command uses MORE in a pipe and filter operation:

dir | find "DIR" | more(cr)

The current directory is piped as input to the FIND command. FIND searches the directory for all DIR entries. The output of FIND is piped as input to MORE. The MORE command stops the directory display when the screen is full. When you press the Return key, another screen of directory entries is displayed.

For more information on filtering data, see Chapter 5.2.

Move or Rename Files (MV)

MV [path]filename.ext [path]filename.ext(cr)

or

MV [path]filename.ext... path(cr)

MV.EXE is an external command that moves files. If you name two files in an MV command, the first file overwrites the second file, and the original file is deleted. If you name a file and a directory, the file is copied into the directory, and the original file is deleted. If you name multiple files and a directory, all the files are moved into the last named directory.

path
: is the path through the subdirectory system to the file(s) you want to move, or to the file or directory where you want to move the file(s).

filename
: In the first parameter, filename is the name of the file(s) you want to move. If the second parameter names a file, the first file is copied into the second file, and the original file is deleted. If the two files are on the same drive, MV renames the first file as the second filename, and deletes the second file if it already exists.

You can use wild-card characters in the first filename if you are moving files into a directory. If you use wild-card characters in the first filename of an MV command moving files into another file, however, MV moves only the first matching file into the destination file.

Then MV displays an error message like this:

```
I already moved d:FILENAME.1 to d:DESTFILE
and will not now move d:FILENAME.3 to d:DESTFILE
```

The following example moves all .EXE files to the \BIN directory on drive B:

```
A>mv  *.exe  b:\bin(cr)

      mv MV.EXE            to (directory) b:\bin
      mv CONCAT.EXE        to (directory) b:\bin
      mv FORMAT.EXE        to (directory) b:\bin
      .    .    .          .     .     .    .
      .    .    .          .     .     .    .
```

This MV command renames MEMO.MEM to SEPT28.MEM:

mv memo.mem sept28.mem(cr)

You can refer to any number of drives in an MV command. For example, this command moves all .EXE files and all files with names starting with SAVE on drive A to the \BACKUP directory on drive B:

mv a:*.exe a:save*.* b:\backup(cr)

Path to Command File (PATH)

PATH [path][;path]...(cr)

PATH is an internal command that searches first the current directory then the other specified directories for command or batch files (.COM, .EXE, or .BAT files **only**). PATH allows you to load external commands and batch files from a subdirectory without typing the full path to the subdirectory each time. PATH without parameters displays the previously specified PATH command.

For example, if you enter this command:

path \cmd(cr)

and then enter:

path(cr)

MS-DOS displays:

```
PATH=\CMD
```

If a PATH command was not previously specified, MS-DOS displays: "No Path".

;

used with PATH and no other parameters tells MS-DOS to search only the current directory for a command or batch file. A semicolon between directory names in the path specifies more than one path (indicated by the \s) for the command or batch file search.

path

is the path through the subdirectory system to the subdirectory containing the external commands and batch files.

Suppose BIN is the first-level subdirectory and contains all the external commands. You can specify:

```
path \bin(cr)
```

Then you can enter any external command without specifying the \BIN subdirectory, regardless of the directory you are using.

If you want MS-DOS to search more than one drive or directory for a command or batch file, you can specify:

```
path a:\bin;a:\bin\utils;c:\pascal\programs;d:\(cr)
```

MS-DOS searches the current directory first, then the PATH directories in the order that they were specified. In the above example, the directory A:\BIN is searched before A:\BIN\UTILS. The first .COM, .EXE, or .BAT file found whose name matches the command entered will be loaded.

If your PATH command contains an invalid delimiter or parameter, such as the wrong drive name, MS-DOS does not detect it until it looks for a command or batch file. Then MS-DOS displays either or both of the following error messages:

```
Invalid drive in search path
Bad command or file name
```

Pause Batch Process (PAUSE)

PAUSE [comment](cr)

PAUSE is an internal command for batch files that suspends batch execution and displays the message: "Strike any key when ready...". Pressing any key except CTRL-C resumes execution of the batch file. Pressing CTRL-C displays:

```
Abort batch job (Y/N)?
```

If you press Y, the remainder of the batch file is not executed, and the MS-DOS command prompt is redisplayed.

At each PAUSE in the file, MS-DOS stops. You can then decide if you want to end processing or do some other action, such as changing diskettes.

comment
: is any string of characters (121 maximum) that you specify. The characters are displayed with the "Strike any key when ready..." message.

For example, this command stops batch file execution:

pause Insert a diskette in drive b(cr)

MS-DOS displays:

```
Insert a diskette in drive b
Strike any key when ready...
```

Print Queued Files (PRINT)

PRINT [drivename:]filename.ext [/C] [/P] [/T] ...(cr)

PRINT is an external command that puts files in a print queue and prints them while MS-DOS is doing other tasks. Only files from the current directory can be put in the print queue.

You cannot specify a path with the PRINT command. Instead, use the CHDIR command to move to the directory that contains the file you want to print. PRINT without any parameters displays the names of the files in the queue.

When you specify PRINT for the first time after MS-DOS starts, MS-DOS displays this message:

```
Name of list device [PRN]:
```

The list device is the printer. Specify the valid assignment name for your list device, such as COM1, COM2, or LPT1 (see the CLST command in this chapter for more information). If you press the Return key, MS-DOS uses PRN as the list device. PRN is a logical device name equivalent to LST.

After you give the device name for your printer, the screen displays:

```
Resident part of PRINT installed
```

If you give an invalid assignment name for the device, MS-DOS displays:

```
List output is not assigned to a device
```

Be sure that the list device is attached to your computer.

The list device cannot be used when PRINT is being processed. MS-DOS displays an error message until the files are all printed or printing is terminated with the /T parameter.

drivename
: is the name of the drive containing the directory of files that you are currently using. If you do not specify a drive, MS-DOS uses the default drive. Do not remove the disk from the drive while the files are being printed. If MS-DOS tries to access the file and the drive is not ready, MS-DOS displays:

```
Drive not ready
```

If MS-DOS finds a disk error, the file being printed is canceled. The printer types a disk error message, advances to the next page, and the printer alarm sounds. MS-DOS then prints the rest of the files in the print queue.

filename
: is the name of the file(s) that you want to print. MS-DOS queues and prints the files in the order you specify in the command. You can specify a maximum of 10 files.

If you enter PRINT and the name of the file(s) without any other parameters, for example:

print chapt1.doc chapt2.doc chapt3.doc(cr)

MS-DOS queues and prints the files. MS-DOS displays the name of the drive and the name of the first file being printed:

```
A:CHAPT1      .DOC is currently being printed
```

The names of the remaining files in the queue are also displayed:

```
CHAPT2      .DOC is in queue
CHAPT3      .DOC is in queue
```

MS-DOS prints a file, moves to the next page, then prints the next file. Tabs are expanded with blanks to the next eighth column.

Wild-card characters can be used in the filename and its extension (see Chapter 2.3).

/C

cancels the named file(s). The filename preceding the /C and subsequent filenames are canceled until MS-DOS finds a /P in the command line. For example, the following command cancels CHAPT1, CHAPT2, and CHAPT3 from the print queue and prints CHAPT4.DOC:

print chapt1.doc /c chapt2.doc chapt3.doc chapt4.doc /p(cr)

This command removes all CHAPT?.??? files from the print queue:

print chapt?.* /c(cr)

If there are no other files in the print queue and a file is printing—for example, MORE.TXT—and you enter:

print more.txt /c(cr)

MS-DOS prints:

```
A:MORE      .TXT Canceled by operator
```

The printer advances one page, and the printer alarm sounds once. MS-DOS then displays on your screen:

```
PRINT queue is empty
```

If you specify:

print more.txt echo.txt(cr)

MS-DOS displays:

```
A:MORE      .TXT is currently being printed
ECHO        .TXT is in queue
```

If you then type:

print more.txt /c(cr)

while the MORE file is printing, MS-DOS:

- ▶ Stops printing the MORE file
- ▶ Prints the operator-canceled message
- ▶ Advances the paper one page
- ▶ Sounds the printer alarm
- ▶ Prints the next file in the queue
- ▶ Displays:

```
A:ECHO    .TXT is currently being printed
```

/P

prints the named file(s). The file preceding the /P and subsequent files are printed until MS-DOS finds a /C in the command line. For example, this command prints CHAPT5 and CHAPT6, cancels CHAPT7, and prints CHAPT8:

print chapt5.doc /p chapt6.doc chapt7.doc /c chapt8.doc /p(cr)

/P is the default if you specify only PRINT and the filename.

/T

deletes all queued files from the print queue. When you use /T, do not specify any other parameters. For example, this command empties the print queue:

print /t(cr)

MS-DOS displays:

```
PRINT queue is empty
```

MS-DOS prints:

```
All files canceled by operator
```

MS-DOS Prompt (PROMPT)

PROMPT [prompt-text](cr)

PROMPT is an internal command that changes the MS-DOS default prompt. PROMPT without parameters designates the default prompt to be A>.

prompt-text
  represents alphanumeric character(s) that you specify.

For example, you can specify the prompt as COMMAND by typing:

prompt COMMAND(cr)

MS-DOS displays the new system prompt instead of the A> prompt:

```
COMMAND
```

If you type the following, the A > prompt returns:

```
COMMANDprompt(cr)
A>_
```

The next example changes the prompt from A > to READY, and then reinstates the A > prompt:

```
A>prompt READY(cr)
READYprompt(cr)
A>_
```

You can insert special characters in the prompt by specifying $c for the prompt-text, where c is:

$	The $ character
t	The current time (set when MS-DOS was loaded or by the last TIME command)
d	The current date (set when MS-DOS was loaded or by the last DATE command)
p	The name of the current directory of the default drive
v	The MS-DOS version number
m	The current operating mode (V or I)
n	The default drive
g	The > character
l	The < character

b	The | character
s	A leading space only
e	The ESC character
_	Carriage return/line feed, which returns to the beginning of the next line on the screen

The special characters inserted with $ must be lowercase. If you specify $ and any other character than the ones described, MS-DOS does not process the command.

To use the current time and date as a system prompt, type:

```
prompt TIME = $t$_DATE = $d(cr)
```

MS-DOS displays, as a two-line command prompt, the current time and date:

```
TIME=10:54:30.50
DATE=Mon 7-04-85
```

Recover Files (RECOVER)

RECOVER [path]filename.ext(cr)

or

RECOVER drivename:(cr)

RECOVER is an external command. Use this command when MS-DOS cannot read a file because of a defective disk. If a file or a diskette directory has a bad sector, this command recovers the file or the entire disk. If there is a bad sector in the directory, this command recovers all the files on the disk.

path
: is the path through the subdirectory system to the file you want to recover.

filename.ext
: is the file that has the bad sector. The file is read sector by sector from the disk. MS-DOS marks the bad sector, skips it, and allocates it to a system table. The data in the bad sector is not recovered. MS-DOS does not make any more file allocations to that sector.

The RECOVER command and the drivename without any parameters recovers all the files on the disk. MS-DOS scans the File Allocation Table (FAT) for chains of allocation units (the amount of space used on a disk for a file). MS-DOS creates a new root directory. Each entry in the root directory is renamed as:

FILEnnnn.REC

where *nnnn* is a sequential number starting with 0001. Each FILEnnnn.REC points to the beginning of an allocation chain.

If the root directory cannot contain all of the allocation unit chains, RECOVER displays a message. The unrecovered chains are left in the File Allocation Table (FAT). Reissue RECOVER and the drive name when you have made more room in the root directory.

Remark (REM)

REM comment(cr)

REM is an internal command for batch files. During batch execution, each REM command displays the comment stored on that command line in the batch file.

comment
: is a string of characters (123 maximum) that you want displayed when batch execution encounters the REM command in the batch file. The only delimiters you can use in the comment are a blank space, tab, or comma:

```
REM This is a file to display the main directory
PAUSE Insert disk in drive B:
DIR B:
```

You can use a period (.) in place of REM to indicate a comment line. For the first line in the preceding example, you can enter:

```
.This is a file to display the main directory
```

Rename (REN)

REN [path]filename.ext filename.ext(cr)

REN is an internal command that renames files.

path
: is the path through the subdirectory system to the directory containing the file you want to rename. Path always includes the drive name if the drive is not the default drive.

filename.ext
: The first filename is the name of the file to be renamed. The second filename is the new name. The file itself remains on the diskette where it was originally created.

You can use wild-card characters in either filename. This command changes all the files with .DAT extension to .CMD extension:

```
ren *.dat *.cmd(cr)
```

The following command renames the file PHNLIST on drive B to PHNNUM. The file remains on drive B:

```
ren b:phnlist phnnum(cr)
```

If you rename a file with a name that is already in the directory, MS-DOS displays:

```
Duplicate filename or File not found.
```

PlusPC System Copy (RETROSYS)

RETROSYS source [/A] destination [/A](cr)

RETROSYS is an external command that copies the MS-DOS system files from one specified directory to another. If the destination is a diskette, the diskette must be in VICTOR format.

Because there is a unique set of files for I mode and one for V mode, RETROSYS can copy one or both sets of files. These files are as follows:

IDOS.SYS
VDOS.SYS
ICONFIG.SYS
VCONFIG.SYS
ICOMMAND.COM
VCOMMAND.COM
ICONFIG.BAT
VCONFIG.BAT

source
: is the path to the directory from which you want to transfer the MS-DOS system files. If you specify only the drive letter, you must include the colon (A:). The source must contain the files ICOMMAND.COM, VCOMMAND.COM, and the operating system files. If you want to use the current default directory as the source, you can enter the following in place of the directory name:

(,)

destination
: is the path to the directory to which you want to transfer the MS-DOS system files. If you specify only the drive letter, you must include the colon (A:). If you do not specify a destination drive letter, RETROSYS asks you for one.

/A

is the command switch that copies ICONFIG.SYS/VCONFIG.SYS and ICONFIG.BAT/VCONFIG.BAT to the destination. Any device drivers specified in ICONFIG.SYS or VCONFIG.SYS are also copied.

The following command copies all the system files from the current directory on drive A to the root directory of drive B:

```
retrosys a: b:(cr)
```

If you are currently in the root directory of drive A, you can enter the following command to accomplish the same system file transfer:

```
retrosys (,) b:(cr)
```

If you want to copy the files ICONFIG.SYS/VCONFIG.SYS and ICONFIG.BAT/VCONFIG.BAT from drive A to drive B, enter:

```
retrosys a:/a b:(cr)
```

Remove Directory (RMDIR)

```
RMDIR [path](cr)
```

RMDIR is an internal command that removes the last named subdirectory in a directory hierarchy. You can abbreviate RMDIR to RD. Chapter 3 gives more information about RMDIR and the subdirectory system.

path

is the path through the directory hierarchy to the subdirectory you want to delete.

RMDIR will not delete a subdirectory if it is not empty. If you try to delete the current directory or a subdirectory containing any entries other than the . and .. (notations indicating this directory level and the next higher level), MS-DOS displays:

```
Invalid path, not directory
or directory not empty
```

Delete (DEL) all files from the subdirectory you want to remove. Delete all directory entries from the subdirectory with the RMDIR command.

You can use RMDIR and the directory name without a backslash to delete a subdirectory entry in the current directory. Suppose you have a directory hierarchy of:

\MGR1.NAM\JOBS\BENEFITS

If you are currently working with the JOBS subdirectory, you can delete BENEFITS from JOBS by entering:

rmdir benefits(cr)

If you are using any directory except BENEFITS, the command:

rmdir \mgr1.nam\jobs\benefits(cr)

removes the BENEFITS subdirectory, if BENEFITS is empty. To remove the JOBS subdirectory, return to the root directory with the CHDIR command, make sure JOBS is empty, and then enter RMDIR with the path to JOBS.

Single Drive Disk Copy (SDCOPY)

(V mode only)

```
SDCOPY [drivename:](cr)
```

SDCOPY is an external command that copies the contents of one diskette onto a VICTOR format diskette in the same disk drive. You insert the source diskette (the diskette to be copied) into the drive, and its contents are read into memory. Then you remove the source diskette and insert a VICTOR formatted destination diskette into the same drive. The contents of the source diskette are written from memory onto the destination diskette. SDCOPY is useful for copying diskettes on a single-drive system. SDCOPY does not format the destination diskette; you must use a formatted diskette.

For more information on formatting a diskette in V mode, see the FORMAT command for V mode in this chapter.

drivename
: is the name of the drive to be used to make the diskette copy.

When you issue the SDCOPY command, screen prompts ask you to insert the source diskette into the drive and then swap the destination diskette. You may have to reinsert the source and destination diskettes more than once depending upon the size of the diskette contents and the amount of available memory.

For example:

```
A>sdcopy(cr)
Single Drive Disk Copy Utility, Ver. x.y
Insert source disk, then press space bar
(return key to exit)(sp)
3 insertions of your source and destination
disks will be required.
Partial read #1 of 3
Insert formatted destination disk, then press
space bar (return key to exit)(sp)
Partial write #1 of 3
Insert source disk, then press space bar
(return key to exit)(sp)
Partial read #2 of 3
Insert formatted destination disk, then press
space bar (return key to exit)(sp)
Partial write #2 of 3
Insert source disk then press space bar (return
key to exit)(sp)
Partial read #3 of 3
Insert formatted destination disk, then press
space bar (return key to exit)(sp)
Partial write #3 of 3
SDCOPY completed
Insert source disk, then press space bar
(return key to exit)(cr)
A>_
```

You can stop the copy process by pressing the Return key in response to a prompt for a diskette. If you stop the copy before it is complete, the destination diskette is unusable.

You can use single- or double-sided diskettes. If you copy a single-sided diskette onto a double-sided diskette, the double-sided diskette becomes single-sided.

Search Files (SEARCH)

SEARCH files or directories [matching constraints] [actions](cr)

SEARCH.EXE is an external command that does these three things:

- ▶ Searches through the specified files and/or directories
- ▶ Selects matching files according to the specified constraints
- ▶ Either lists the matching file(s) or performs the specified action(s) on the matching file(s)

You specify the action you want SEARCH to perform with command switches (such as /CP for copy files). With SEARCH you can:

- ▶ Move, copy, or delete files
- ▶ Archive files by creating a disk archive file
- ▶ Change file attributes (such as read-only or read/write status)
- ▶ Find the sum of file sizes
- ▶ Delete empty subdirectories

Additional switches allow you to request prompting, exclude subdirectories from the search, and copy entire tree structures (subdirectory hierarchies).

Because SEARCH has several types of parameters, the syntax description for this utility is divided into sections listing and describing each type of parameter: *Files or Directories*, *Matching Constraints*, *Actions*, and *Action and Constraint Modifiers.*

The first section presents the syntax for "files or directories" and gives examples showing how to use file and directory names with SEARCH. In the other three sections, the command switches for that set of parameters are presented in table form. Examples showing all the SEARCH command parameters are given in a final section, *Hints for Using SEARCH.*

Files or Directories—The Domain

The first set of SEARCH parameters—files or directories—immediately follows the keyword SEARCH in a command. This parameter list consists of one or more subdirectory names and/or filenames which can include the MS-DOS wild-card characters. The list is the possible domain from which SEARCH selects files to operate on. If only a drive name is given, SEARCH begins at the current working directory of that drive and searches downward through any subdirectories below. To display the names of the current working directories, enter CD /A.

If you want SEARCH always to start at the root directory in the domain, use a \ at the beginning the path. For example, if you specify:

```
search a:\ b:\ other parameters(cr)
```

the SEARCH utility searches downward from the root directories on drives A and B only. Without the \, the search is relative; in other words, the search starts at the current working directory, which might not be the root directory.

You can mix filenames and directories to specify a domain of some specific files and the files in some subdirectories. This command:

```
search b:\rtc tod.src tod.asm other parameters(cr)
```

considers the file or directory B:\RTC, and any subdirectories below it, and the two TOD files on the default drive when executing the other parameters.

Matching Constraints

The second set of (optional) parameters in a SEARCH command is matching constraints. These are command switches that further define which files in the specified domain of files and/or directories you want to affect with SEARCH. Only files from the domain that meet all the conditions listed as matching constraints are acted on. You can only use each of the switches listed in Table 6-1 once in a command line.

Table 6-1: Matching Constraint Switches for SEARCH

SWITCH	EFFECT
/NAME filenames	Selects files from the domain that match the filenames following /NAME. To list more than one filename, separate them with exclamation marks, like this: **/name rtc.asm!tod.asm!tod.obj(cr)** Details about including filenames with SEARCH are given in the paragraphs following this table.
/^NAME filenames	Selects files from the domain that do NOT match any of the names following /^NAME. See the examples following this table.
/FNAME pathname	Selects files whose full pathnames, including subdirectories and filename, match the names following /FNAME. See the examples following this table for a description of what the pathnames can be.
/^FNAME pathname	Selects files whose full pathnames do not match any of the names following /^FNAME. See the examples following this table.
/ATTR attribute expression	Selects files whose attributes make the specified attribute expression true. The expression consists of letters representing one of the five "file attributes" described here. Multiple letters can be separated with & (for AND), ! (for OR), and ^ (for unary NOT). You can also use parentheses for grouping subexpressions. The five file attribute letters are: A = archive status is set on (i.e., the file has been edited since the last archive) D = this "file" is a subdirectory H = this file is a hidden file R = this file is read-only S = this file is a system file To select read/write files (files that are not read-only) with archive status set on: **search a:\ /attr a&^r(cr)** To display the status of the five file attributes for your files, use the LS /L command (or SEARCH with the /LS action switch).

SWITCH	EFFECT
/COMP directoryname	Compares the files in the domain with files of the same name in the specified subdirectory. Selects only those files that have the same name and are equivalent to a file in the specified directory. Any differences between the two files are displayed. Include the /T switch with /COMP to preserve the directory tree structure of the domain. With /T, the target directory name for the compare is computed as follows: 1. The drive prefix (and any .\ or ..\ prefixes) are deleted from the directory name of the file found (i.e., the drive name is ignored). 2. The remainder of the directory name including the filename from the domain is appended to the directory named after the /COMP to produce the file which is to be compared. For example, SEARCH A: /COMP TEST /T compares files on drive A with files in the relative path TEST. If the file A:\X\Y exists, SEARCH compares it to TEST\X\Y but not to TEST\Y. SEARCH A: /COMP TEST without the /T switch compares A:\X\Y to TEST\Y.
/^COMP directoryname	Selects files from the domain if there is not a file of the same name in the specified path OR (if there is a file of the same name) the two files are not equivalent. Any differences between the two files are displayed. The description of /COMP gives more details.
/EXISTS directoryname	Selects files from the domain whose names match filenames in the specified subdirectory. If /T is used, the path is computed as for /COMP.
/^EXISTS directoryname	Selects files from the domain whose names do not match any filenames in the specified path. If /T is included, the path is computed as for /COMP.
/LE filename	Selects files from the domain whose time of creation (date and time) precedes or is the same as that of the named file. Read LE as "less than or equal to."
/LT filename	Selects files whose time of creation (date and time) precedes that of the named file. Read LT as "less than."
/GE filename	Selects files whose time of creation (date and time) follows or is the same as that of the named file. Read GE as "greater than or equal to."

SWITCH	EFFECT
/GT filename	Selects files whose time of creation (date and time) follows that of the named file. Read GT as "greater than."
/LEDATE filename	Selects files whose date of creation (regardless of time) precedes or is the same as that of the named file. Read LE as "less than or equal to."
/LTDATE filename	Selects files whose date of creation (regardless of time) precedes that of the named file. Read LT as "less than."
/GEDATE filename	Selects files whose date of creation (regardless of time) follows or is the same as that of the named file. Read GE as "greater than or equal to."
/GTDATE filename	Selects files whose date of creation (regardless of time) follows that of the named file. Read GT as "greater than."
/SIZE n	Selects files that are at least *n* bytes in size.
/SIZEB n	Selects files that are at least *n* (512-byte) blocks in size.

File and directory names listed with the /NAME, /^NAME, /FNAME, and /^FNAME switches can have several types of wild-card characters, some of which are themselves switches. The wild cards you can use are as follows:

- The MS-DOS ? wild card matches any single character, or none.
- The * matches any number of characters anywhere in the text. In most MS-DOS commands, *A* is equivalent to * because the first * pads to the end of the name. With the /NAME switch, however, *A* matches any text that contains an A.
- The /. switch makes wild cards also match the file extension separator (the . between the filename and extension). For example, without the /. switch, you must specify *.* to match all files; using * with the /. switch matches all files.
- The /\ switch makes wild cards also match the \ in a pathname. For example, A**.PAS matches A\X\Y.PAS and A\Z\W.PAS, but not A\X\Y\Z\Q\R\X.PAS. Using the * with the /\ switch also matches this last path.

- ▶ Multiple-choice selection. If you enclose characters in brackets, SEARCH looks for only the characters listed. For example, [AEIOU]*.ASM matches any filename beginning with a vowel and having the extension .ASM.

For example, these two /NAME or /^NAME phrases are equivalent:

```
[bv][tc][1k]fd*.*(cr)
[bv][tc][1k]FD* /.(cr)
```

Both match any file with B or V as its first character, T or C as its second, 1 or K as its third, and FD as its fourth and fifth characters. The second example uses the /. switch to match the . between the name and extension. Note that upper- and lowercase do not matter.

The next example matches any file in the root directory on drive A or in any subdirectory on drive A which does not have the extension .ASM:

```
search a:\ /^name *.asm(cr)
```

The next example matches any file in drive A's root directory or any subdirectory on A whose name does not begin with B or V:

```
search a:\ /^name [bv]*.*(cr)
```

The last example uses both /NAME and /^NAME. The files selected from the domain, which is the root directory of A, must have the extension .ASM but cannot begin with BT1:

```
search a:\ /name *.asm /^name BT1*.*(cr)
```

The difference between the /FNAME and /NAME switches is that /NAME performs the matching test only on the file's name, while /FNAME performs its test against the full pathname for that file, starting at the drive name and the root directory.

For example, the following command selects filenames beginning with SRC and having no filename extension in the root directory on drive B, or any subdirectory on drive B that begins with SRC.

```
search b:\ /fname b:\src*(cr)
```

Adding the /\ switch matches any subdirectories below the first one. Adding /. matches any file extension. For example, this command matches files in any path beginning with B on drive A:

```
search a:\ /fname a:\b* /\ /.(cr)
```

To match drive designators, the format is D:\. For example:

```
search a:\ /fname a:\t* /\ /.(cr)
```

selects files from the root directory on drive A that begin with T, or any files in subdirectories on A whose directory names begin with T.

The last example selects all files from drive A that are not in subdirectory A:\TEMP:

```
search a:\ /^fname a:\temp\* /\ /.(cr)
```

Actions

The third set of parameters for SEARCH are action switches. These options specify the function to be performed on the files that meet the matching constraints. With action switches you can set and reset file attributes; move, copy, delete, and back up files; print directory entries; and find the sum of the file sizes.

Table 6-2 lists and describes the switch settings for actions. Any action marked with an asterisk can be specified only once in a SEARCH command.

Table 6-2: Action Switches for SEARCH

SWITCH	EFFECT
/ATTR + attrs	Sets the specified file attributes on. Specify "attrs" as single-character attributes, as for the /ATTR matching constraint (see Table 6-1). Do not use separators between attribute letters. For example, this command sets archive and read-only status on: **/attr + ar(cr)**
/ATTR- attrs	Set the specified attributes off.
/BACKUP directoryname *	Is an abbreviation for: **/cp directoryname /attr a /attr- a(cr)** That is, this switch finds archivable files (those with archive status) and resets archive status off after the copy. To preserve the tree structure of the files you back up, use the /T switch. You cannot use /BACKUP with /CP.
/CP directoryname *	Copies the matching files into the specified directory. This cannot be used with /BACKUP. You can use the /T option as for the /COMP matching constraint switch (see Table 6-1).
/LS	Lists a directory of matching files in long format. /LS displays file attribute settings, file size, time and date of creation, and the full pathname. If an attribute is not set for a file, / is printed; if it is set, the attribute is printed. For read/write files, however, **w** is printed instead of the /. See /ATTR in Table 6-1 for a list of the attributes.
/MV directoryname *	Moves the matching files from the domain into the specified subdirectory. The original files are deleted. If the /T option is specified, the tree structure of the source file is preserved. For example, this command: **search a: /mv z /t(cr)** finds A:\X\Y; the file Z\X\Y would be created rather than Z\Y (the file created without /T).

SWITCH	EFFECT
	The destination directory name is formed from the source pathname as follows: 1. The drive prefix and any .\ or ..\ is removed from the source directory name. 2. The remainder of the source directory name is appended to the directory name after the specified /MV directory name.
/PRINT	Lists the matching files from the domain. PRINT is the default action for SEARCH.
/RM	Removes (deletes) the matching files. If you specify both /CP and /RM, the copy is done before the removal. If you specify /MV with /RM, /RM has no effect.
/RMDIR	Removes any empty directories encountered. If /MV is specified, the directory remove is attempted after the files are moved. This command moves all of X to drive B, and then removes all of subdirectory X: **search x /mv b: /rmdir(cr)**
/SUM	Displays the sum of the sizes of matching files. This is the exact size of the data in the files and does not consider allocation units' round-off.
/TARA filename.ext *	Appends the matching files to an existing TAR (tape archive) file. Otherwise acts same as /TARC.
/TARC filename.ext *	Creates a TAR (tape archive) file named FILENAME.EXT containing all matching files. For example, this command puts all .PAS files on the default drive into the file PASBKUP.TAR: **search *.pas /tarc pasbkup.tar(cr)**
/TARX filename.ext *	Extracts the matching files from the tape archive file FILENAME.EXT. The only action you can perform with this option is to /LS, /PRINT, /SUM, or /CP files. The /RM or /MV switches are ignored. This command lists the files in the tape archive file PASBKUP.TAR: **search /tarx pasbkup.tar /ls(cr)** This command copies the files in the given TAR file into directory XYZ on the default drive: **search /tarx pasbkup.tar /cp xyz(cr)**

*These switches can appear only once in a SEARCH command.

Action and Constraint Modifiers

You can use the switches listed in Table 6-3 to modify both the matching constraints and the actions listed in a SEARCH command. You can group single-letter options after a single switch signal (such as /AV) providing they do not form one of the multiple-letter options. Each multiple-letter option must have a separate signal. For example, you cannot combine the /AV switch with the /NR switch to make /NRAV.

Table 6-3: SEARCH Action and Constraint Modifiers

SWITCH	EFFECT
/A	Matches only files with the archive status on. Short for /ATTR A.
/N	Takes no actions. SEARCH displays what it would have done. The /N switch is equivalent to using /P and always responding negatively.
/NR	Does no recursive searching of subdirectories. SEARCH normally searches any directories you specify and all subdirectories. Using /NR limits searches to only the directories specified.
/P	Prompts (asks yes/no) for /CP, /MV, /RM, /RMDIR, /TARA, /TARC, /TARX, /BACKUP, /ATTR-, or /ATTR + options. Use this switch to verify SEARCH's actions before you take an action that would have a permanent or destructive effect.
/S	Is silent (displays nothing).
/T	Preserves tree structure of directories for /MV, /CP, /COMP, /^COMP, /EXISTS, /^EXISTS, /TARX, and /BACKUP options.
/V	Preserves volume/drive structure. An option for /TARX only, /V returns files from tape archive to their original volume or drive.
/.	Allows the file extension separator (.) to be wild-carded.
/\	Allows the directory separator (\) to be wild-carded.

Hints for Using SEARCH

- ▶ Remember that all matching constraints must be met (they are ANDed together) before a file is processed (listed, deleted, moved, or archived) by SEARCH.
- ▶ You can mix and match any constraints, action switches, and modifying options switches.
- ▶ If you are doing anything that might cause data to be lost or reformatted, add the /P switch to prompt you for verification of the action, or append the /N switch to display the actions that would occur if you execute the command.

Use the following list of examples as guidelines when you use the SEARCH command:

1. If you specify only a drive name in the domain, such as:

   ```
   search f: /name *.pas(cr)
   ```

   SEARCH does not necessarily search all files on that drive. SEARCH starts at the current working directory and skips any higher directories. To search drive F for files with the .PAS extension, either do a CD F:\ before the preceding command, or use:

   ```
   search f:\ /name *.pas(cr)
   ```

2. To list all the directories on drive D, type:

   ```
   search d:\ /attr d(cr)
   ```

3. To copy all files that have been modified since the last time you did a backup, use:

   ```
   search a:\ /backup b:(cr)
   ```

   This copies modified files to drive B and resets their archive status off.

If the archive status of your files is unknown, enter:

```
search a:\ /attr+ a(cr)
```

to set archive status on for all files (so all files will be archived).

4. Because some commands are abbreviations, remember that there are some restrictions. For example,

   ```
   search a: /backup b: /attr r(cr)
   ```

   is illegal because /BACKUP B: is an abbreviation for /CP B: /ATTR A /ATTR- A, and only one /ATTR is allowed. Instead you should enter:

   ```
   search a: /cp b: /attr a&r /attr- a(cr)
   ```

5. To find all files created on the same day as SOFTWARE.MEM, on drives A and C, use:

   ```
   search a:\ c:\ /ledate software.mem /gedate software.mem(cr)
   ```

   This command lists all files with dates less than or equal to AND greater than or equal to SOFTWARE.MEM's date. Thus, all files created on that date will be listed.

   To copy all of these files to drive B's working directory, enter:

   ```
   search a:\ c:\ /ledate software.mem /gedate software.mem
      /cp b:(cr)
   ```

6. To list all hidden and system files on drives/volumes A, C, D, and E, enter:

   ```
   search a:\ c:\ d:\ e:\ /attr s!h /ls(cr)
   ```

7. Before you archive files with SEARCH /TARC, you can see the amount of space that will be occupied by the archived files. If you are about to archive drives/volumes A, C, and D, enter:

   ```
   search a:\ c:\ d:\ /a /sum(cr)
   ```

The results tell you the size of the tape archive file that will be created when these files are archived.

8. To archive the files from drives/volumes A, C, and D, enter:

   ```
   search a:\ c:\ d:\ /a /tarc b:backup /attr- a(cr)
   ```

   This command finds all files with archive status on (/A), archives the files into a single file called BACKUP on drive B (/TARC B:BACKUP), and then resets archive status off (ATTR- A).

9. To re-arrange your directory structure, you can use SEARCH to copy entire subdirectories. For example, to move the subdirectory A:\COMMAND\SOURCES and all of its subdirectory structures, including all files, to C, use:

   ```
   search a:\command\sources /mv c:\ /t(cr)
   ```

   In this example, using the /T option preserves the tree structure (\COMMAND\SOURCES\...). Without the /T switch, all the files of all the directories would be copied into the current working directory of C.

10. To remove any empty directories on drives E and F, use:

    ```
    search e:\ f:\ /rmdir(cr)
    ```

    The /RMDIR switch can be used with any other actions. Only empty directories will be removed; also, any drive's current working directory cannot be removed.

11. If there are duplicate files in the two directories F:\PROJECTS and G:\MYPROJ, you can delete the redundant files from G:\MYPROJ with this command:

    ```
    search g:\myproj /exists f:\projects /rm(cr)
    ```

    If you want to delete only files that contain the same data as the similarly named files in F:\PROJECTS, use the following:

    ```
    search g:\myproj /comp f:\projects /rm(cr)
    ```

Set Equivalent Value (SET)

SET [string1 = [string2]](cr)

SET is an internal command that sets one string value equivalent to another string. In subsequent MS-DOS or application program operations, the equivalent value is substituted for the first named string.

With SET, you change the operating environment. The operating environment is a series of names and parameters built by the command processor and passed to all programs that it invokes. You can SET almost anything into the environment. The only syntax requirement is that a single equals sign (=) be in the command line.

Entering SET without any parameters displays all the values that have been set with the SET command. For example:

set(cr)

lists all the values currently set, such as:

```
path=
prompt=$n$g
```

string1
is an environment-name or variable name that you want to set equivalent to the value of string2. For example, you can duplicate the function of the PROMPT command with SET. Enter:

set prompt = YES?(cr)

to change the system prompt to YES?.

To clear any setting, enter the SET command giving only the first string and an equals sign:

```
set prompt = (cr)
```

string2
: is the value that you want to substitute for string1 in subsequent operations.

SET is useful if you are doing batch file processing or using an application program that repeats a variable. For example, in a batch file you can create replaceable parameters with names instead of the usual numbers. Replaceable parameters defined as names must have a leading and trailing %, such as %FILE%. You can assign the value to use for that parameter with the SET command:

```
set file = domore(cr)
```

When MS-DOS processes the batch file, %FILE% becomes the filename DOMORE. Using this procedure, you do not have to edit a file to change the parameter names before you run the batch file.

Shift Replaceable Parameters (SHIFT)

SHIFT(cr)

SHIFT is an internal command for batch files that moves all replaceable parameters one position to the left:

%1 replaces %0

.

.

.

%10 replaces %9

When there are more than 10 parameters on a command line, each additional parameter shifts into %9.

For example, this batch file named DIRLOOK.BAT uses the SHIFT command to display as many directories as you enter when you load the batch file:

```
dir /p %1
:nexdir
shift
if %1  ==  stop goto lastone
dir /p %1
goto nexdir
:lastone
cd \
```

If you specify:

```
dirlook managers scheds stop(cr)
```

MS-DOS searches the root directory for the MANAGERS and the SCHEDS directories. During batch processing, MS-DOS displays each batch command (except REM lines or lines beginning with colons) and then performs the command on the line.

MANAGERS is used for %1. Because of the SHIFT command, %1 becomes SCHEDS. PROJ then replaces SCHEDS, and so on. The batch file completes when STOP equals STOP in the IF command. MS-DOS displays:

```
A>dir /p managers
```

Next the MANAGERS directory is displayed.

```
A>shift

A>if scheds == stop goto lastone

A>dir /p scheds
```

Next the SCHEDS directory is displayed.

```
A>goto nexdir

A>shift

A>if stop == stop goto lastone

A>cd \
```

The CD \ at the end of the file returns you to the root directory. See Chapter 3 for information on changing directories, or see CHDIR in this chapter.

Sort Data (SORT)

[command |] SORT [/R] [/+n] [< [drivename:]input source]
[> [drivename:]output source](cr)

SORT.EXE is an external command that filters data by rearranging it alphanumerically (using the ASCII collating sequence—see Appendix A). The data to be rearranged can be the output from an MS-DOS command.

The command can be any external MS-DOS command whose output you want to sort. The | symbol tells MS-DOS to direct (pipe) the output from the MS-DOS command to the SORT command, described in Chapter 5.3.

For example, this command sorts the current directory alphabetically:

```
dir | sort(cr)
```

The command:

```
dir | find "< DIR >" | sort(cr)
```

uses the output from the DIR command as input to the FIND command. The output of FIND, the names of all the subdirectories in the current directory, is sorted alphabetically and displayed on the screen.

/R
tells MS-DOS to reverse the alphanumerical sort (Z to A, or *n* to 0 where *n* is an integer).

/+n
tells MS-DOS to sort the data by columns. *n* is the number of the column where the SORT is to start. For example, this command:

```
dir | sort /+1(cr)
```

sorts the directory alphanumerically by filename, which is the first column in the directory display.

This command does a reverse alphanumeric sort of the directory:

```
dir | sort /r /+1(cr)
```

<

tells MS-DOS to use the parameters following the < symbol as input.

drivename

is the name of the drive containing either the input data to be sorted or the sorted output data. The drive name is optional when the input or output source is on the default drive.

input source

is the data you want to sort. It can be a disk file or data from a MODEM. For a disk file, specify the name of the file and the path to the file if it is not in the current directory. For a MODEM or any other device, you must specify the physical device name.

>

tells MS-DOS to use the parameters following the > symbol as output. You must include an output source if you include the > symbol. For example, if you enter:

```
dir | sort >(cr)
```

MS-DOS displays:

```
File creation error
```

output source

is the destination of the sorted data from the input source. It can be files, printers, or other devices. For files you must include the path to the file unless it is in the current directory. This command:

dir | sort > lst(cr)

sorts the directory alphabetically and sends the output to the LST device, the printer. The default output is > CON. If you do not include an output source, the output from the SORT command is sent to CON, the CRT screen.

The command:

find "95066" mail.lst | sort > zip.cod(cr)

pulls out all the records containing the zipcode 95066 from the MAIL.LST file and sorts them alphabetically into the ZIP.COD file. ZIP.COD is created by this command, but if ZIP.COD already exists, its contents are overwritten by the output of SORT. To see the sorted data records in ZIP.COD, enter:

type zip.cod(cr)

MS-DOS displays this message, followed by an alphabetical list of each 95066 record:

```
---------- mail.lst
```

Print End of File (TAIL)

```
TAIL [/n] [files or directories](cr)
```

TAIL.EXE is an external command that displays the last n lines of a file or list of files.

/n

is the number of lines at the end of the file(s) you want to display onscreen. If you do not include a number, TAIL prints the last 10 lines. The maximum value is approximately 450. If a line exceeds 130 characters, TAIL truncates it.

files or directories

is a list of subdirectory names and/or filenames to be operated upon. If you do not include files or directories, TAIL gets its input from the keyboard or from redirected input.

For example, this command lists the last 3 lines of every file with .LST extension in the current directory:

```
tail /3 *.lst(cr)
```

Either of the next two commands lists the last 10 lines of the file ACCOUNTS. The < symbol in the first command indicates that ACCOUNTS is being redirected as input to the TAIL command.

```
tail < accounts(cr)
tail accounts(cr)
```

The following command prints the last 50 lines of every file in the \TEMP subdirectory on drive A:

```
tail /50 a:\temp(cr)
```

TIME

TIME [hh][:mm][:ss](cr)

TIME is an internal command that sets the time for the operating system. If you enter the command without parameters, the current time is displayed and you are prompted for a new time. Press the Return key if you do not want to change the time. For example:

```
A>time(cr)
Current time is 13:01:10.00
Enter new time:9:05(cr)
```

The new time known to the operating system is now 09:05:00.00. The colon separates the hours, minutes, and seconds. The period separates the hundredths of a second.

6

hh
: represents the hour. Use 00 to 23.

mm
: represents the minutes. Use 00 to 59.

ss
: represents the seconds. Use 00 to 59. You do not have to type the seconds or hundredths of a second.

If you enter the TIME command with valid parameters (numbers, not letters, in the indicated range for each parameter), MS-DOS displays the system prompt.

If the parameters are invalid, such as letters instead of numbers or a larger than acceptable range of numbers, MS-DOS displays:

```
Invalid time
Enter new time:_
```

MS-DOS also displays this message if you use invalid delimiters, such as a hyphen instead of a colon or a period.

List Directory Paths (TREE)

TREE [drivename:] [/F](cr)

TREE is an external command that lists all directory paths found on the specified drive. TREE can also list all the files in each directory. If you do not specify a drive letter with the TREE command, the default drive is used.

drivename:
: is the name of the drive for which you want to list all the directory paths.

/F
: lists the names of the files in each subdirectory.

If you enter the TREE command without any parameters, you might see a listing on your screen similar to the following:

```
C>tree(cr)

DIRECTORY PATHS FOR VOLUME ............ VOLUMENAME

Path:     C:\CAM

Sub-directories:  -none-

Path:     C:\TEMP

Sub-directories:  -none-

Path:     C:\LC

Sub-directories:  VICTOR
                  SOURCE
```

If you enter the TREE command with the /F option, the preceding listing is expanded to include all the files in each subdirectory.

```
C>tree/f(cr)
DIRECTORY PATHS FOR VOLUME ............ VOLUMENAME

Path:      C:\CAM
Sub-directories:   -none-

     Files:        CHAPT1.TXT
                   CHAPT2.TXT
                   CHAPT3.TXT
                   APPA.TXT
                   APPB.TXT

Path:      C:\TEMP

Sub-directories:   -none-

     Files:        LTRFORM.LIB
                   MEMOFORM.LIB
                   CALVIN.LTR

Path:      C:\LC

Sub-directories:   VICTOR
                   SOURCE

     Files:        VICTOR.BAT
                   MAIN.OBJ
                   HOMER.BAT
```

Type File (TYPE)

```
TYPE [path]filename.ext(cr)
```

TYPE is an internal command that displays the contents of text files. You cannot use TYPE with .COM or .EXE files because they contain non-alphanumeric characters, such as Control or Escape sequences. Tabs expand to 8 characters (column 8, 16, and so on through column 80).

path
: is the path through the subdirectory system to the file you want to display.

filename.ext
: is the name of the file to be displayed.

For example, to display the contents of the JOBS.LST file in the \MGR1.NAM\JOBS subdirectory, enter:

```
type \mgr1.nam\jobs\jobs.lst(cr)
```

Reload Command Processor (VCOMMAND)

VCOMMAND is an external command that invokes the command processor, VCOMMAND.COM. See the description of ICOMMAND in this chapter.

Version of (VER)

VER(cr)

VER is an internal command that displays the version of MS-DOS that you are using. For example, the following command:

ver(cr)

displays:

```
MS-DOS Version  x.y
```

where *x.y* is the version number of the operating system.

Verify Write to Disk (VERIFY)

VERIFY [ON](cr)

or

VERIFY [OFF](cr)

VERIFY is an internal command that verifies whether subsequent data written by MS-DOS onto the disk can be read by MS-DOS when requested. VERIFY has the same function as the V switch on the COPY command. VERIFY without parameters displays the current setting (ON or OFF) of the command. For example, if you type:

verify(cr)

MS-DOS displays:

```
VERIFY is xx
```

where *xx* is "on" or "off."

ON
: tells MS-DOS to verify the data each time it writes data to the disk. MS-DOS issues an error message if it cannot do the write operation. Note that when VERIFY is ON, any command involving a write to the disk performs slower because of the extra time involved to verify the operation.

OFF
: turns the verify instruction off.

Volume Label Display (VOL)

VOL [drivename:][/C](cr)

VOL is an internal command that displays the Volume ID of a diskette or fixed (hard) disk volume. A Volume ID is a name or label you give to a diskette during formatting, or to a fixed (hard) disk volume during initialization. A Volume ID can be up to 11 characters long, using the valid characters for filenames. MS-DOS displays the Volume ID for a diskette or volume when you enter a DIR command, a CHKDSK command, or a VOL command.

drivename
: is the name of the drive whose ID you want to check. If you do not specify a drive, MS-DOS displays the ID for the default drive.

/C
: is a switch you use to request a change in the Volume ID.

If you try to display the Volume ID for a diskette or volume that does not have an ID, MS-DOS displays:

```
Volume ID for drive x: has not been set.
```

where *x* is the name of the drive.

If you type:

vol b: /c(cr)

MS-DOS displays the Volume ID for drive B, and prompts you for a new Volume ID:

```
Volume ID for drive B: is ARCHIVE1031
Enter new Volume ID:_
```

After you enter a new Volume ID and press Return, MS-DOS writes the new ID to the disk. You can display the new Volume ID for drive B by entering:

vol b:(cr)

Word and Line Count (WC)

WC [/R] [/W](cr)

or

WC [/R] [/W] [path...](cr)

or

WC [/R] [/W] [filename...](cr)

WC.EXE is an external command that counts words and/or lines in a file or files. WC can count all the files in a directory or in a hierarchy of directories. WC lists the number of lines, the number of words, the filename, and the total count of lines and words. Entering WC with no parameters counts the lines and words in the keyboard input (or redirected input) that follows, up to end-of-file (CTRL-Z Return).

/R
: is a command switch that expands the count to include files in all subdirectories below the directory named in the WC command. To use /R, you must list a directory, not a file, in your WC command.

  The following example uses /R with the root directory to count the words and lines in all the files in every directory on the default drive:

```
A>wc /r \(cr)
      21 L        386 W   \VMS-DOS.SYS
      43 L        495 W   \VCOMMAND.COM
       5 L         16 W   \VCONFIG.SYS
      49 L        297 W   \SUBDIR1\FINAL.MEM
      86 L        455 W   \SUBDIR2\CHAP2.DOC
      . .         .   .   .   .   .
      . .         .   .   .   .   .
     563 L       6511 W   total
```

/W
: is a command switch that omits the word count from the WC operation. Counting only lines is faster than counting words and lines.

path
: is the path through the subdirectory system to the file or subdirectory you want to count. If you list a directory name (or the single \ to indicate the root directory) without a filename, WC counts all the files in that directory.

filename
: is the name of the file you want to count. You can use wild-card characters to refer to more than one file.

The first example below counts lines and words in the FINAL.MEM file, and the second example counts lines in the FINAL.MEM and FINAL.BAK files:

```
wc final.mem(cr)

wc /w final.mem  final.bak(cr)
```

You can list multiple files and directories in a WC command, and the files and directories can be on different drives. The following command counts the number of lines in all files with .DAT extension on the default drive as well as all files in the subdirectory \DRAFT1 on drive B:

```
wc /w *.dat b:\draft1(cr)
```

6

MS-DOS Messages

This chapter lists many of the commonly generated messages you might see while working at your computer. The messages are boldface and listed alphabetically. An explanation follows each message. Sometimes you are given a reason for why you are seeing that message and what you can do to successfully complete the procedure you want. The word(s) at the beginning of the explanation, preceding the colon, indicate which program you were trying to execute when you received the message.

Some error messages are generated by a command program that resides in both I mode and V mode. Although the two command programs are different for each mode, you resolve the error in the same manner (independent of which mode you are in). For ease of reference, the cause of an error message is listed "generically" instead of by mode. For example, error messages shown to be generated by:

COMMAND.COM	are generated by ICOMMAND.COM if you are in I mode or VCOMMAND.COM if you are in V mode.
CONFIG.SYS	are generated by ICONFIG.SYS in I mode or by VCONFIG.SYS if you are in V mode.
DOS.SYS or MS-DOS	are generated by IDOS.SYS in I mode or by VDOS.SYS if you are in V mode.

If a message pertains to only one mode, there is an explanation in the text following that message.

When MS-DOS encounters a problem, it displays an error message on the screen. Different error messages are displayed for different types of errors. Error messages are displayed in situations like these:

▶ You have entered a command that MS-DOS does not recognize.

▶ You have misspelled a filename, or you have entered an incorrect drive name.

▶ You have inserted a diskette that does not contain the file or program that MS-DOS is looking for.

If you get a floppy disk error when you try to load MS-DOS from diskette, remove the diskette to escape from the error situation. Check that the diskette is a system diskette and that it is not write-protected. Try inserting it again or use another system diskette.

If you see the word "filename" or "xxxx" in brackets or parentheses in one of the messages, it means that the message you see on your screen contains the specific filename or number(s) that are pertinent to the operation you are attempting.

^C

MS-DOS: MS-DOS has detected a CTRL-C (03H) from the console device.

Abort edit (Y/N)?

EDLIN: This message is displayed when you choose the Q (Quit) command in EDLIN. The Quit command exits the editing session without saving any editing changes. Specify Y for yes or N for no.

Abort, Retry, Ignore?

COMMAND, MS-DOS: If a disk or device error occurs at any time during a command or program, MS-DOS returns this message and asks you to abort the command or program, retry it, or ignore the error.

All files cancelled by operator

PRINT: This message is displayed when you specify the /T switch with the PRINT command.

All specified files are contiguous

CHKDSK: All files are allocated contiguously on the disk without fragmentation.

Allocation error for file (filename)

CHKDSK: The named file had a data block allocated to it that does not exist (that is, a data block number larger than the largest possible block number). CHKDSK truncates the file short of the bad block.

Allocation error in file, size adjusted

CHKDSK: An invalid sector number was found in the FAT. The file was truncated at the end of the last valid sector.

Are you sure (Y/N)?

COMMAND: MS-DOS displays this message if you try to delete *.* (all files in the current directory) or if you try to delete a directory (DEL directoryname). Specify Y (for Yes) or N (for No).

Bad call format reading drive (x:)

Device error: See Appendix B.

Bad call format writing drive (x:)

Device error: See Appendix B.

Bad command error reading drive (x:)

Device error: See Appendix B.

Bad command error writing drive (x:)

Device error: See Appendix B.

Bad command or file name

COMMAND: The command processor cannot find the file you asked it to run. You either mistyped the filename or the file does not exist on the disk. Check to see that you are in the correct directory and that you have specified the path correctly.

Bad destination (must be a drive).

RETROSYS: You have not specified the destination drive in correct notation, d:.

Bad DISKCOPY command

DISKCOPY: You entered the command incorrectly; retype the command.

BAD MEDIA. PRESS RETURN KEY TO START OVER OR ALT-C TO EXIT.

SDCOPY: Utility encountered a bad sector while reading or writing a diskette. Reformat or replace the diskette.

Bad or missing (filename)

MS-DOS: You specified an invalid device in the CONFIG.SYS file. Check the accuracy of the DEVICE statement in the CONFIG.SYS file.

Bad or missing Command Interpreter

MS-DOS: MS-DOS either cannot find or cannot read the COMMAND.COM file on the disk; either the file is missing from the root directory, or the file is invalid. Restart the system, or copy COMMAND.COM from your system disk onto the disk that you use to start MS-DOS. You can also receive this message if COMMAND.COM has been moved from the directory it was originally in when you started MS-DOS. When MS-DOS cannot find or read the command interpreter specified in CONFIG.SYS, MS-DOS defaults to either A:ICOMMAND.COM or A:VCOMMAND.COM. You should always load the correct version of COMMAND.COM.

Bad parameter on command line.

RETROSYS: You have used an invalid parameter on the command line. Correct the command. You might have not specified a parameter, or you might have specified an invalid drive, an invalid destination, or too many parameters.

Bad source (must be drive or file name).

RETROSYS: You have not specified the source drive in correct notation, d: or [path]filename.ext.

Bad unit error reading drive (x:)

Device error: See Appendix B.

Bad unit error writing drive (x:)

Device error: See Appendix B.

Bad volume ID

COMMAND, DIR, VOL: You tried to list the directory on a disk with a bad Volume ID.

Batch file missing

COMMAND: MS-DOS cannot locate the batch file as specified in the CONFIG.SYS file. You have tried to execute a batch file from another batch file, but MS-DOS cannot find the batch file as specified.

BREAK is off (or on)

BREAK, COMMAND: This message tells you the current setting of BREAK.

Cannot CHDIR to (filename) - tree past this point not processed

CHKDSK: CHKDSK is traversing the tree structure of the directory and is unable to proceed to the specified directory. Subdirectories beneath this directory are not verified.

Cannot CHDIR to root
Processing cannot continue

CHKDSK: CHKDSK is traversing the tree structure of the directory and is unable to return to the root directory. CHKDSK cannot continue checking the remaining subdirectories to the root.

Cannot do binary reads from a device

COMMAND: This message appears during COPY command processing. The COPY cannot be done in binary mode when you are copying from a device. Remove the /B switch or specify an ASCII copy with the /A switch.

Cannot edit .BAK file—rename file

EDLIN: You attempted to edit a backup copy created by EDLIN. Either rename the file or copy the .BAK file and give it a different extension.

CANNOT FORMAT TRACK

FORMAT: You might have a bad diskette. Run FORMAT again. If the error persists, discard the diskette.

Cannot open (filename)

PRINT: Either MS-DOS cannot find the specified file to print or the file does not exist. Check the command for a valid filename.

CANNOT READ TRACK

DISKCOPY: You might have a bad diskette. Run the program again. If the error persists, discard the diskette.

Cannot recover . entry, processing continued

CHKDSK: The . entry (current directory) is defective.

Cannot recover .. entry

CHKDSK: The .. entry (parent directory) is defective.

CANNOT SIZE TRACK

DISKCOPY, FORMAT: You might have a bad diskette. Run the program again. If the error persists, discard the diskette.

Cannot write label on drive

RETROSYS: The diskette is write-protected, or you might have a bad diskette. Remove the write-protect tab, or insert a different diskette.

CHDIR .. failed, trying alternate method

CHKDSK: In traveling the tree structure, CHKDSK was not able to return to a parent directory. It will try to return to that directory by starting over at the root and traveling down.

CLOSE DRIVE DOOR

SDCOPY: The drive door was open during an attempted read or write of sector.

Command not found

COMMAND: You typed an incorrect command or an invalid delimiter, parameter, or filename. Retype the command or filename correctly.

Content of destination lost before copy

COPY: A file to be used as a source file to the COPY command has been overwritten before completion of the copy. During file concatenation (COPY with the + parameter), MS-DOS found a source filename with the same name as the destination filename. MS-DOS skips the source file and displays this message. For example, if you enter this command, MS-DOS destroys FILE2 before it can be copied:

```
copy file1 + file2 file2(cr)
```

Convert lost chains to files (Y/N)?

CHKDSK: If you respond Y to this prompt, CHKDSK recovers the lost blocks it found when checking the disk. CHKDSK creates a directory entry and a file for you with the filename FILEnnnn. If you respond N, CHKDSK frees the lost blocks so they can be reallocated. If you did not specify the /F switch in the command line, you can still receive this message, but CHKDSK will not be able to convert the lost chains.

Copy aborted at track nn

DISKCOPY: The diskette might have a bad track, and the program is unable to continue. Run the program again, or try moving to another drive. If the error persists, discard the diskette.

Copy another (Y/N)?

DISKCOPY: Respond Y if you want to copy another diskette. Respond N if you do not want to copy another diskette.

Copy complete

DISKCOPY: DISKCOPY has completed processing.

Copy completed.

RETROSYS: This message appears when your system files are successfully copied to the destination drive.

Copy failed.

RETROSYS: This message appears when your system files have not been successfully copied to the destination drive.

Copy not completed

DISKCOPY: DISKCOPY could not copy the entire disk.

Copy to B: (FLOPPY). Press space bar when ready.

RETROSYS: You see this message when you are using a dual-floppy drive system and you have specified only the source of the system files in the command line. Press the space bar to start the system copy.

Copying...

DISKCOPY: This message indicates that DISKCOPY is copying a diskette.

Copying configuration file:

RETROSYS: You see this message as RETROSYS copies the CONFIG.SYS and CONFIG.BAT files.

Copying device drivers:

RETROSYS: You see this message on the screen as RETROSYS copies the device drivers to the specified drive.

Copying system files:

RETROSYS: You see this message as the system files are being copied.

< CR > for retry, or new drive =

COMMAND: COMMAND.COM must be reloaded, but the PlusPC cannot find the file. Insert a diskette with COMMAND.COM in the current drive and press Return, or enter the path to the directory where COMMAND.COM is located and press Return.

Current date is (mm-dd-yy)

COMMAND, DATE: This message is displayed in response to the DATE command. If the date is correct, press Return; if not, enter a new date.

Current time is (hh:mm:ss:hh)

COMMAND, TIME: This message is displayed in response to the TIME command. If the time is correct, press Return; if not, enter the correct time.

Data error reading drive (x:)

Device error: See Appendix B.

Data error writing drive (x:)

Device error: See Appendix B.

Destination drive? Press RETURN to end.

RETROSYS: After designating the destination drive letter, press the Return key to complete this prompt.

Destination FAT read failed.

RETROSYS: There is a disk read error. Remove the diskette and reinsert it in the drive and try again, or use a different diskette.

Directory error - file: (filename)

CHKDSK: No valid data blocks are allocated to the named file. CHKDSK deletes the file.

Directory is totally empty, no . or .. files

CHKDSK: The specified directory does not contain references to current and parent directories. Delete the specified directory and recreate it.

Disk error reading drive (x:)

Device error: See Appendix B.

Disk error reading FAT (x)

CHKDSK: One of your File Allocation Tables has a defective sector in it. MS-DOS automatically uses the other FAT, but it is a good idea to copy all your files to another disk.

DISK error while reading drive A
Abort, Ignore, Retry?

DISKCOPY: MS-DOS encountered disk errors during DISKCOPY processing. Refer to Appendix B, "Device I/O Errors," for more information on this message.

Disk error writing drive (x:)

Device error: See Appendix B.

Disk error writing FAT (x:)

CHKDSK: One of your File Allocation Tables has a defective sector in it. MS-DOS automatically uses the other FAT. It is a good idea to copy all your files onto another disk.

Disk full—write not completed

EDLIN: You gave the END command, but the disk did not contain enough free space for the file. EDLIN aborted the E command and returned you to the MS-DOS prompt. You should remove some files from the disk that is full, or copy the file to another disk. Part of the file might have been written to disk and saved. Delete the saved portion and restart the editing session. The file will not be available after this error.

Disk not initialized

CHKDSK: No directory or file allocation table was found. If files exist on the disk, and the disk has been physically harmed, you might still be able to transfer files from this disk to recover data.

Disk unsuitable for system drive

FORMAT: FORMAT detected a bad track on the disk where system files should reside. Use another disk for your system files and throw away the bad disk.

Disks must be the same size

DISKCOPY: You cannot copy the contents of a disk with a different format using DISKCOPY. Use the COPY command to copy files onto the disk.

Divide overflow

MS-DOS: The 8088 has set the divide overflow flag which is usually caused by division by zero. MS-DOS has received an overflow interrupt (or a rampant program has simulated the interrupt). Restart your system.

(.)(..) Does not exist

CHKDSK: Either the . or .. directory entry is invalid.

DOUBLE-SIDED SOURCE DISK CANNOT BE COPIED TO SINGLE-SIDED DESTINATION.

SDCOPY: You have tried to copy a double-sided diskette to a single-sided diskette. SDCOPY will again prompt you to insert a destination diskette. Replace the destination diskette with a double-sided one.

Drive = n Track = n Error = MM
Abort, Retry, Ignore?

Type A (to abort) the command, I (to ignore) the error, or R (to retry) the command that generated this error.

DRIVE DOOR OPENED

DISKCOPY, FORMAT: Close the drive door and restart the program.

Drive not ready

PRINT: During a print operation, you removed the diskette containing the file being printed.

Duplicate file name or File not found

COMMAND, RENAME: You have tried to rename a file to a filename that already exists, or the name you specified could not be found.

ECHO is off (or on)

COMMAND: This message tells you the current status of ECHO.

End of input file

EDLIN: The entire file was read into memory. If the file is read in sections, this message indicates that the last section of the file is in memory.

Enter new date:

COMMAND, DATE: You must respond to this prompt when you start MS-DOS. Enter the date in the mm/dd/yy format.

Enter new time:

COMMAND, TIME: You must respond to this prompt when you start MS-DOS. Enter the time in the hh:mm:ss format.

Enter new volume ID:

COMMAND, VOL: Respond to this prompt by entering a volume ID for the disk. A Volume ID can be up to eleven characters long, using valid characters for filenames. You see this message only when you use the /C switch with VOL.

Entry error

FOR: The %% parameters specified in the FOR command must match: FOR %%A IN (%%02 %%03 %%04) DO MKDIR %%A.

EDLIN: The last command you entered contained a syntax error. Retype the command with the correct syntax and press Return.

Entry has a bad attribute (or link or size)

CHKDSK: This message can be preceded by one or two periods indicating which subdirectory is invalid. If you have specified the /F switch, CHKDSK attempts to correct the error.

Error in .EXE file

COMMAND: The .EXE file you tried to load has an invalid internal format.

(type of error) error reading file

Device error: See Appendix B.

ERROR WHILE READING ON DRIVE x
Abort,Ignore,Retry:

Type A (abort), I (ignore), or R (retry). Retry the operation before ignoring or terminating (aborting) the program. Ignoring the error could result in bad data. Check the resulting file carefully.

ERROR WHILE WRITING ON DRIVE x
Abort,Ignore,Retry:

Type A (abort), I (ignore), or R (retry). Retry the operation before ignoring or terminating (aborting) the program. Ignoring the error could result in bad data. Check the resulting file carefully.

Error writing to device

COMMAND: You tried to send too much data to a device. MS-DOS was unable to write the data to the specified device.

Errors found, F parameter not specified

CHKDSK: Corrections will not be written to disk. CHKDSK found errors on the disk. If you have not specified the /F switch, CHKDSK continues printing messages, but does not correct the errors.

Errors on list device indicate that it might be offline. Please check it.

PRINT: Your printer is off-line. Check that your printer is turned on, that there is paper in it, and that the cables are connected.

EXEC failure

COMMAND: Either MS-DOS found an error when reading a command or the FILES statement in the CONFIG.SYS file is set too low. Increase the value and reboot MS-DOS.

File allocation table bad

COMMAND: The disk might be defective. Run CHKDSK to check the disk.

File allocation table bad drive (x:)

CHKDSK: The disk might be defective. Run CHKDSK to check the disk.

FILE ALLOCATION TABLE BAD FOR DRIVE x

Check that the diskette is formatted. The message indicates that the copy in memory of one of the allocation tables has pointers to nonexistent blocks.

File cannot be converted

EXE2BIN: The input file is not in the correct format. CS:IP does not meet the criteria specified, or it meets the .COM file criterion but has segment fixups. This message is also displayed if the file is not a valid executable file.

File cannot be copied onto itself
0 File(s) copied

COMMAND, COPY: The source filename you specified is the same as the destination filename. For example, you entered:

```
copy file1 file1(cr)
```

The file copy must be renamed if it is on the same drive, or it must be on a different drive than the source file.

File Creation Error

SORT: You used the directional output symbol (>) in the SORT command, but did not specify an output source. Or, MS-DOS could not complete the copy of a fixed disk volume to a diskette because the volume directory was larger than the diskette directory.

File creation error

CHKDSK, COMMAND, EXE2BIN: With CHKDSK, you tried to add a new filename or replace a file that already exists in the directory. If the file already exists, it is a read-only file and cannot be replaced. Run CHKDSK on the disk to determine the cause of the error. With EXE2BIN, EXE2BIN cannot create the output file. Run CHKDSK to determine if the directory is full, or if some other condition caused the error.

File size error for file (filename)

CHKDSK: The size of the file in its directory differs from its actual size. CHKDSK adjusts the directory to indicate the actual file size. The amount of useful data might be less than the size shown because the last data block might not be used fully.

(filename) contains non-contiguous blocks

CHKDSK: The filename specified is not allocated contiguously on the disk. If you specify the /F switch, CHKDSK can fix this error.

(filename) file not found

PRINT: You switched disks while a file was queued up, but before it started to print. Reissue the PRINT command for that filename.

(filename) is cross linked on cluster

CHKDSK: Make a copy of the file you want to keep, and then delete both files that are cross linked.

(filename) is currently being printed

PRINT: The filename specified is being printed.

(filename) is in queue

PRINT: The filename specified is waiting to be printed.

Filename must be specified

EDLIN: You did not specify a filename when you started EDLIN.

File not found

COMMAND, EXE2BIN, FIND, EDLIN, RECOVER: MS-DOS cannot find the file you specified. Check to see that the pathname is accurate and that the file exists in the directory you specified.

Files cross-linked: (filename) and (filename)

CHKDSK: The same data block is allocated to both files. CHKDSK does not try to correct this. To correct the problem, first COPY both files; then delete the originals. Review each file for validity and edit as necessary.

File not found xxxx.xxx

FIND: where *x* is a character in the filename. You entered the FIND command for a string of characters in a file. MS-DOS could not locate the file, because the file is not in your current directory. You need to simplify a path through the directory hierarchy to the subdirectory containing the correct filename.

First cluster number is invalid, entry truncated

CHKDSK: The file directory entry contains an invalid pointer to the data area. If you specified the /F switch, the file is truncated to a zero-length file.

Fixups needed - base segment (hex:)

EXE2BIN: The source (.EXE) file contained information indicating that a load segment is required for the file. Specify the absolute segment address at which the finished module is to be located.

FOR cannot be nested

COMMAND: Nesting FOR statements is not allowed in a batch file.

Format aborted at track nn

FORMAT: The diskette has a bad track, and the program is unable to continue. Run the program again, or try moving to another drive. If the error persists, discard the diskette.

Format another (Y/N)?

FORMAT: Type Y (for Yes) to format another disk. Type N (for No) if you do not want to format another disk. If you accidentally type Y, you can abort the format process by typing CTRL-BREAK in response to the "Strike any key to begin formatting" message.

Format failure

FORMAT: MS-DOS could not format the diskette. This message is always displayed with an explanation as to why MS-DOS could not format the diskette.

Incorrect MS-DOS version

ASSIGN, CHKDSK, EDLIN, FIND, FORMAT, MORE, PRINT, RECOVER, SORT: Many utilities cannot run on earlier versions of MS-DOS. Some of the utilities can run only under the version of MS-DOS for which they were configured. Do not use an older version of CHKDSK because it can delete files.

Insert diskette for drive (x:) and strike any key when ready

MS-DOS: This message appears when MS-DOS is copying and formatting. You should insert a disk in the appropriate drive and press any alphanumeric key to begin processing.

Insert diskette with batch file and press any key when ready

COMMAND: The disk containing the batch file in progress is no longer in the drive in which you started it. Reinsert the disk that contains the batch file in the appropriate drive, or press CTRL-C.

Insert formatted target diskette into drive (x:)

DISKCOPY: DISKCOPY is ready for a disk in the destination drive.

Insert new diskette for drive (x:) and strike any key when ready

FORMAT: Insert a blank disk into the appropriate drive and press any alphanumeric key to begin formatting. **CAUTION:** If there is any data on the disk, it will be destroyed by the format process.

Insert source diskette into drive (x:)

DISKCOPY: Insert the disk to be copied into the specified drive.

Insert target diskette into drive (x:)

DISKCOPY: You are running DISKCOPY and your source and destination drives are the same. Reinsert the destination disk into the specified drive.

Insufficient disk space

COMMAND, EXE2BIN, SORT: The disk is full. It does not contain enough room to perform the specified operation. MS-DOS copied as many files as possible, and there are more files to be copied. Remove the diskette that is full, insert another formatted diskette, and copy the remaining files.

Insufficient memory.

CHKDSK, EDLIN, EXE2BIN, SORT, RETROSYS: There is not enough free memory to perform the specified operation. Reboot and try again, or add more RAM.

Insufficient room in root directory. Erase files in root and repeat CHKDSK

CHKDSK: CHKDSK always recovers lost files into the root directory. In this case, your root directory is full. Delete some files in your root directory to make room for the lost files.

Intermediate file error during pipe

COMMAND: The pipe operation makes use of temporary files on the disk which will automatically be deleted after the piping process is complete. An error has occurred in one of these files.

Invalid baud rate

MODE: You specified an invalid baud rate.

Invalid characters in volume label

FORMAT: The volume label should contain only up to 11 alphanumeric characters.

Invalid characters per line

MODE: You specified invalid characters in this line.

Invalid COMMAND.COM
Insert COMMAND.COM disk in drive (x:) and strike any key when ready

COMMAND: MS-DOS needs to reload the ICOMMAND.COM or VCOMMAND.COM file from disk; however, MS-DOS cannot find ICOMMAND.COM or VCOMMAND.COM on the disk, or the copy found is invalid. Insert a disk into the specified drive which contains a copy of ICOMMAND.COM or VCOMMAND.COM identical to the version on the disk with which you started MS-DOS.

Invalid country code

MS-DOS: You have specified a country number in your CONFIG.SYS file which is not in the table of files configured in this implementation of MS-DOS. Country codes must be in the range 1–99. Consult the documentation on CONFIG.SYS for more information on supported countries.

Invalid current directory

CHKDSK: Your disk is bad. Replace the disk or make another copy from your backup system disk.

Invalid date

COMMAND, DATE: You specified an invalid date in response to the date prompt when starting MS-DOS.

Invalid device

COMMAND: The device specified was not CON, NUL, AUX, LST, or PRN.

Invalid device name

MODE: You specified an invalid device name.

Invalid directory

COMMAND: The directory you specified either does not exist or is invalid. Check to see that you entered the directory name correctly.

Invalid drive in search path

COMMAND: One of the drives specified in the last PATH or SET PATH commands does not exist.

Invalid drive or file name

EDLIN, RECOVER: Specify a valid drive or a valid filename.

Invalid drive specification

CHKDSK, COMMAND, DISKCOPY, FORMAT: Specify a valid drive.

Invalid lines per inch

MODE: You specified an incorrect number of lines per inch. See MODE in Chapter 6.4 for information on how to specify this parameter.

Invalid number of data bits

MODE: You specified an invalid number of data bits. See MODE in Chapter 6.4 for information on how to specify this parameter.

Invalid number of parameters

COMMAND, FIND, MODE, RECOVER: You have specified the wrong number of options in the command line, or you have put an incorrect number or spaces between parameters.

Invalid number of stop bits

MODE: You specified an invalid number of stop bits. See MODE in Chapter 6.4 for information on how to specify this parameter.

Invalid parameter

ASSIGN, CHKDSK, COMMAND, EDLIN, FIND, FORMAT, MODE, PRINT: One of the switches that you have specified is wrong, or you have put an incorrect number or spaces between parameters. In ASSIGN, this message is generated when you specify a drive letter that is not supported by MS-DOS. With MODE, you might have specified a parameter that is not supported in V mode.

Invalid parity

MODE: You specified an invalid parity value. See MODE in Chapter 6.4 for information on how to specify this parameter.

Invalid path or file name

COMMAND: Specify a valid pathname or filename to the COPY, DEL, or TYPE command.

Invalid path, not directory, or directory not empty

COMMAND, RMDIR: You are unable to remove the directory requested for one of the specified reasons.

Invalid screen mode

MODE: You specified an invalid screen mode. The sector size is too large in the file.

Invalid sub-directory entry

CHKDSK: The subdirectory that you specified either does not exist or is invalid. Check to see that you entered the subdirectory name correctly.

Invalid time

COMMAND, TIME: You specified an invalid time in response to the time prompt when starting MS-DOS.

Label not found

COMMAND: There is a GOTO command to a nonexistent label in a batch file.

Line too long

EDLIN: During a Replace command, the string given as the replacement caused the line to expand beyond 253 characters. Divide the long line into two lines and retry the Replace command.

List output is not assigned to a device

PRINT: When you first run PRINT, it asks you what device you want to specify as a print spooler. This message appears if PRINT is set up for a nonexistent device.

Memory allocation error. Cannot load COMMAND, system halted

COMMAND: Restart MS-DOS. The program you just ran might have been written using poor memory allocation techniques that are not acceptable in this version of MS-DOS. If this error persists, make a new copy of the MS-DOS disk from your backup copy of the system disk.

--More--

MORE: Press the Space bar to view more of the file or directory.

Must specify destination line number

EDLIN: You must specify a destination line number when you are copying and inserting lines with EDLIN.

Must specify ON or OFF

BREAK, COMMAND, VERIFY: The command requires an argument, either ON or OFF.

Name of list device [PRN]:

PRINT: This prompt appears when you run PRINT for the first time in any computer session. You can specify any valid device which then becomes the PRINT output device.

New file

EDLIN: This message is printed if EDLIN does not find a file with the name you specified. If you are creating a new file, ignore this message. If you do not intend to create a new file, check to see that you correctly typed the filename of the file you want to edit.

NO DISKETTE OR BAD TRACK

FORMAT: If you have not inserted a diskette in the drive to be formatted, do so and restart the program. If the drive contains a diskette, take it out and retry with another diskette.

No files match (d:filename)

PRINT: A filespec was given for files to add to the queue, but no files match the specification.

No free file handles

COMMAND: Restart MS-DOS. If this message persists, increase the FILES parameter in the CONFIG.SYS file.

No paper error writing device (dev)

Device error: See Appendix B.

No path

COMMAND, PATH: There is no current command search path.

No room in directory for file

EDLIN: You have tried to save a file to the root directory but it is full. Subdirectories are not limited in size as is the root directory.

No room in environment for LOAD

COMMAND: There are too many values set in the current environment to load a program. Use the SET command to set some of the values to null strings.

Non-MS-DOS disk error reading drive (x)

Device error: See Appendix B.

Not enough room to merge the entire file

EDLIN: There was not enough room in memory to hold the file during a Transfer command. You must free some memory by writing some files to disk or deleting some files before you can transfer this file.

Not found

EDLIN: You have specified a Search or a Replace command that was unable to find a further occurrence of the specified Search or Replace string.

Not ready error reading drive (x:)

Device error: See Appendix B.

Not ready error writing drive (x:)

Device error: See Appendix B.

O.K.?

EDLIN: This prompt occurs during Search and Replace command processing. If you press any key except Y or Return, the search or replace continues.

Opening system files:

RETROSYS: You see this message when the system files are being opened on the source diskette.

Operating system mismatch

RETROSYS: This message appears when you try to copy a different version of MS-DOS than RETROSYS is configured for.

Operating system version mismatch

FORMAT: The operating system being used is not compatible with the version of FORMAT being used. Use the correct version of the program or load the correct operating system.

O/S must be MSDOS 2.0 or above.

RETROSYS: You have attempted to copy a version of MS-DOS that is older than version 2.0. RETROSYS runs only under MS-DOS version 2.X or above.

Out of environment space

COMMAND: There is not enough room in the program environment to accept more data.

Path specified is too long for environment

COMMAND: The path specified in a command-line parameter or argument is too long to fit into the environment. Use shorter subdirectory names, fewer levels of subdirectories, or make more room in the environment by removing some set values. (See the SET command in Chapter 6.4.)

Preparing destination

RETROSYS: You see this message when the system files are being written to the destination diskette.

Press any key to begin formatting (x:)

FORMAT: This prompt is issued before you format a disk. Press any alphanumeric key to begin formatting. To discontinue this operation, press CTRL-BREAK.

Press any key to begin recovery of the (xxx) file(s) on drive (x:)

RECOVER: This prompt is issued before you recover a disk or file. Press any alphanumeric key to begin recovering. To discontinue this operation, press CTRL-BREAK.

Press any key when ready

DISKCOPY: This prompt occurs when you are copying disks. When you have inserted the disks into the appropriate drives, press any alphanumeric key to begin copying the disk. To discontinue this operation, press CTRL-BREAK.

PRINT queue is empty

PRINT: There are no files waiting to be printed.

PRINT queue is full

PRINT: There is only room for 10 files in the list of files waiting to be printed.

Printer error.

MODE: The print device generated an error when output was sent to it.

Probable non-DOS disk
Continue (Y/N)?

CHKDSK: The disk you are using is not recognized by this version of MS-DOS. The disk either was created by another system with a format that is not supported on this version of MS-DOS or is not an MS-DOS disk. Do not continue processing if CHKDSK has returned this message for a floppy disk. If this message is returned for a fixed disk, the information describing the disk characteristics to MS-DOS has been destroyed. In this case, continue CHKDSK processing. You will probably have to set up your fixed disk again.

Processing cannot continue

CHKDSK: There is not enough memory free in your machine to process CHKDSK for this disk. You must obtain more memory to run CHKDSK.

Program too big to fit in memory

COMMAND: You must acquire more memory to run your application. Some applications you have run might still be using some memory. You can try to restart MS-DOS; however, if you receive this message again, you must acquire more memory.

Read error: D = nn, T = nn, S = nn, E = nn
CANNOT READ TRACK

DISKCOPY, FORMAT: Retry the program; discard the diskette if the error persists. (D stands for drive, T for track, S for sector, and E for error code.)

Read fault error reading drive (x:)

Device error: See Appendix B.

Resident part of PRINT installed

PRINT: This is the first message that MS-DOS displays when you issue the PRINT command. It means that available memory has been reduced by several thousand bytes to process the PRINT command concurrent with other processes.

Sector not found error reading drive (x:)

Device error: See Appendix B.

Sector not found error writing drive (x:)

Device error: See Appendix B.

Sector size too large in file (filename)

Device error: The specified device driver loaded by CONFIG.SYS uses a sector size larger than that of any other device driver on the system. You cannot run this device driver.

Sector write error: D = nn, T = nn, S = nn, E = nn
CANNOT WRITE DISK LABEL

DISKCOPY, FORMAT: The program is unable to write the disk label to the diskette. The disk label is part of the internal identification system for diskettes. The location and type of the problem are indicated by the drive, track, sector, and error code numbers.

Seek error reading drive (x:)

Device error: See Appendix B.

Seek error writing drive (x:)

Device error: See Appendix B.

Soft format error: D = nn, T = nn, S = nn, E = nn

FORMAT: A soft format error has occurred. The drive, track, sector, and error code numbers are indicated. If there are more than 9 soft errors on a track, the program aborts. Soft errors in moderate numbers do not hurt diskette performance.

Source and destination drives must differ

RETROSYS: You cannot specify the same source and destination drives. You cannot copy the system onto itself.

Source and target diskettes are not the same format. Cannot do the copy.

DISKCOPY: You must have the same size and kind of disks to run DISKCOPY. You cannot copy from a single-sided disk to a double-sided disk. Reformat the target disk so that it is the same format as the source disk.

Specified COMMAND search directory bad

MS-DOS: The SHELL statement in the CONFIG.SYS file is incorrect. The location that you have told MS-DOS to find COMMAND.COM does not exist, COMMAND.COM is not in that place, or you have specified the wrong name.

specify SOURCE of system files.

RETROSYS: You have not specified the source of the system files, so MS-DOS cannot find them.

Strike a key when ready...

COMMAND: This prompt occurs during command processing and is usually accompanied by another message. This message is also displayed if you have inserted a PAUSE statement in a batch file. Usually, you are asked to insert disks into appropriate drives before this prompt. Press any key to begin command processing.

Syntax error

COMMAND, FIND: Make sure you typed the command correctly.

Terminate batch job (Y/N)?

COMMAND: If you press CTRL-BREAK while in batch mode, MS-DOS asks you whether or not you want to end batch processing. Press Y to end processing; press N to continue the batch job.

Track 0 bad - disk unusable

FORMAT: FORMAT can accommodate for defective sectors on the disk except for those near the beginning. FORMAT cannot write the boot record. Use another disk.

Unable to create directory

COMMAND: MS-DOS could not create the directory you specified. You might have a file with the same name, or the disk might be full.

Unrecognized command in CONFIG.SYS

MS-DOS: There is an invalid statement in your ICONFIG.SYS or VCONFIG.SYS file. Check to see that you have not mistyped any statements in the ICONFIG.SYS or VCONFIG.SYS file.

Unrecoverable error in directory
Convert directory to file (Y/N)?

CHKDSK: If you specified /F in the command line, you can respond Y to this prompt and CHKDSK will convert the bad directory into a file. You can then fix the directory yourself or delete it.

VERIFY is off (or on)

COMMAND, VERIFY: This message tells you the current setting of VERIFY.

Volume ID not changed—file of same name exists.

COMMAND, VOL: The volume ID specified after a VOL /C command is the same as a filename that already exists on the volume. Choose another name.

VOLUME IN DRIVE IS NONEXISTENT OR IS NOT A FLOPPY.

SDCOPY: There is no floppy in the disk drive.

Volume in drive (x:) has no label

COMMAND, DIR, VOL: MS-DOS displays this message in response to the DIR or VOL command when you have not specified a volume label.

Volume in drive (x:) is (volume ID)

COMMAND: MS-DOS displays this message in response to the DIR or VOL command when you have specified a volume label.

Volume label (11 characters, ENTER for none)?

FORMAT, VOL: This message is displayed when you specify the /V switch in the FORMAT command or the /C switch in the VOL command. Specify a volume label or press Return to indicate that you do not want a volume label for this disk.

Warning - directory full

RECOVER: The root directory is too full for RECOVER processing. Delete some files in the root directory to free space.

7

WARNING - DOUBLE-SIDED DESTINATION DISK WILL BECOME SINGLE-SIDED AS A RESULT OF THIS OPERATION. DO YOU WISH TO CONTINUE? (y/n)

SDCOPY: You have tried to copy a single-sided diskette to a double-sided diskette. Type y for yes, or n for no.

Warning: Read error in EXE file.

EXE2BIN: The amount read was less than the size of the header. This is a warning message only. EXE2BIN will attempt to continue processing.

Write fault error writing drive (x:)

Device error: See Appendix B.

WRITE PROTECTED DISKETTE

FORMAT: The diskette being formatted has a write-protect tab on it. Remove the tab and try again. **CAUTION:** Files already on the diskette will be erased if you do this.

Write protect error writing drive (x:)

Device error: See Appendix B.

Write protect violation

FORMAT: The disk is write-protected.

WRONG DESTINATION DISK

SDCOPY: The current destination diskette is different from the original one. This message appears in multiple insertions of the destination diskette. Insert the correct destination diskette.

Wrong destination label type

RETROSYS: The PlusPC operating system can only be copied to a VICTOR-formatted diskette or a PlusPC-configured fixed (hard) disk.

WRONG SOURCE DISK

SDCOPY: The current source diskette is different from the original one. This message applies to multiple insertions of the source diskette. Insert the correct source diskette.

(xxxx) of (xxxx) bytes recovered

RECOVER: This message tells you how many bytes MS-DOS was able to recover of the disk or file.

(filename) cancelled by operator

PRINT: This message is printed on the printer when you specify the /T switch in the PRINT command.

The following messages are displayed when certain MS-DOS commands have completed processing. These are informational messages only.

(xxxx) bytes available on disk

CHKDSK, FORMAT

(xxxx) bytes disk space freed

CHKDSK

(xxxx) bytes disk space would be freed

CHKDSK

(xxxx) bytes free

COMMAND

(xxxx) bytes in (yyyy) directories

CHKDSK

(xxxx) bytes in (yyyy) hidden files

CHKDSK

(xxxx) bytes in (yyyy) recovered files

CHKDSK

(xxxx) bytes in (yyyy) user files

CHKDSK

(xxxx) bytes in bad sectors

FORMAT

(xxxx) bytes total disk space

FORMAT

(xxxx) bytes total memory

CHKDSK

(xxxx) bytes used by system

FORMAT

(xxxx) bytes would be in (yyyy) recovered files

CHKDSK

(xxxx) file(s)

COMMAND, DIR

(xxxx) file(s) copied

COMMAND, COPY

(xxxx) file(s) recovered

RECOVER

(xxxx) lost clusters found in (xxxx) chains

CHKDSK

(filename) contains (xxxx) non-contiguous blocks

CHKDSK

(filename) has invalid cluster, file truncated

CHKDSK

Volume (filename) created (mm, dd, yyyy)

CHKDSK

A

ASCII Character Codes

ASCII Value		Screen	Code	
Decimal	Hex	Character	Name	Keystroke
000	00	(null)	NUL	—
001	01	☺	SOH	CTRL-A
002	02	●	STX	CTRL-B
003	03	♥	ETX	CTRL-C
004	04	♦	EOT	CTRL-D
005	05	♣	ENQ	CTRL-E
006	06	♠	ACK	CTRL-F
007	07	(beep)	BEL	CTRL-G
008	08	◘	BS	CTRL-H
009	09	(tab)	HT	CTRL-I
010	0A	(line feed)	LF	CTRL-J
011	0B	(home)	VT	CTRL-K
012	0C	(form feed)	FF	CTRL-L
013	0D	(carriage return)	CR	CTRL-M
014	0E	♫	SO	CTRL-N
015	0F	☼	SI	CTRL-O
016	10	►	DLE	CTRL-P
017	11	◄	DC1	CTRL-Q
018	12	↕	DC2	CTRL-R
019	13	‼	DC3	CTRL-S
020	14	¶	DC4	CTRL-T
021	15	§	NAK	CTRL-U
022	16	▬	SYN	CTRL-V
023	17	↨	ETB	CTRL-W
024	18	↑	CAN	CTRL-X
025	19	↓	EM	CTRL-Y
026	1A	→	SUB	CTRL-Z
027	1B	←	ESC	Escape Key
028	1C	(cursor right)	FS	
029	1D	(cursor left)	GS	
030	1E	(cursor up)	RS	
031	1F	(cursor down)	US	

ASCII Value		Screen	ASCII Value		Screen
Decimal	Hex	Character	Decimal	Hex	Character
032	20	(space)	064	40	@
033	21	!	065	41	A
034	22	"	066	42	B
035	23	#	067	43	C
036	24	$	068	44	D
037	25	%	069	45	E
038	26	&	070	46	F
039	27	'	071	47	G
040	28	(	072	48	H
041	29	)	073	49	I
042	2A	*	074	4A	J
043	2B	+	075	4B	K
044	2C	,	076	4C	L
045	2D	-	077	4D	M
046	2E	.	078	4E	N
047	2F	/	079	4F	O
048	30	0	080	50	P
049	31	1	081	51	Q
050	32	2	082	52	R
051	33	3	083	53	S
052	34	4	084	54	T
053	35	5	085	55	U
054	36	6	086	56	V
055	37	7	087	57	W
056	38	8	088	58	X
057	39	9	089	59	Y
058	3A	:	090	5A	Z
059	3B	;	091	5B	[
060	3C	<	092	5C	\
061	3D	=	093	5D	]
062	3E	>	094	5E	^
063	3F	?	095	5F	_

ASCII Value		Screen	ASCII Value		Screen
Decimal	Hex	Character	Decimal	Hex	Character
096	60	`	128	80	Ç
097	61	a	129	81	ü
098	62	b	130	82	é
099	63	c	131	83	â
100	64	d	132	84	ä
101	65	e	133	85	à
102	66	f	134	86	å
103	67	g	135	87	ç
104	68	h	136	88	ê
105	69	i	137	89	ë
106	6A	j	138	8A	è
107	6B	k	139	8B	ï
108	6C	l	140	8C	î
109	6D	m	141	8D	ì
110	6E	n	142	8E	Ä
111	6F	o	143	8F	Å
112	70	p	144	90	É
113	71	q	145	91	æ
114	72	r	146	92	Æ
115	73	s	147	93	ô
116	74	t	148	94	ö
117	75	u	149	95	ò
118	76	v	150	96	û
119	77	w	151	97	ù
120	78	x	152	98	ÿ
121	79	y	153	99	Ö
122	7A	z	154	9A	Ü
123	7B	{	155	9B	¢
124	7C	¦	156	9C	£
125	7D	}	157	9D	¥
126	7E	~	158	9E	Pt
127	7F	⌂	159	9F	ƒ

ASCII Value		Screen	ASCII Value		Screen
Decimal	Hex	Character	Decimal	Hex	Character
160	A0	á	192	C0	└
161	A1	í	193	C1	┴
162	A2	ó	194	C2	┬
163	A3	ú	195	C3	├
164	A4	ñ	196	C4	─
165	A5	Ñ	197	C5	┼
166	A6	ª	198	C6	╞
167	A7	º	199	C7	╟
168	A8	¿	200	C8	╚
169	A9	⌐	201	C9	╔
170	AA	¬	202	CA	╩
171	AB	½	203	CB	╦
172	AC	¼	204	CC	╠
173	AD	¡	205	CD	═
174	AE	«	206	CE	╬
175	AF	»	207	CF	╧
176	B0	░	208	D0	╨
177	B1	▒	209	D1	╤
178	B2	▓	210	D2	╥
179	B3	│	211	D3	╙
180	B4	┤	212	D4	╘
181	B5	╡	213	D5	╒
182	B6	╢	214	D6	╓
183	B7	╖	215	D7	╫
184	B8	╕	216	D8	╪
185	B9	╣	217	D9	┘
186	BA	║	218	DA	┌
187	BB	╗	219	DB	█
188	BC	╝	220	DC	▄
189	BD	╜	221	DD	▌
190	BE	╛	222	DE	▐
191	BF	┐	223	DF	▀

ASCII Value		Screen
Decimal	Hex	Character
224	E0	α
225	E1	β
226	E2	Γ
227	E3	π
228	E4	Σ
229	E5	σ
230	E6	μ
231	E7	τ
232	E8	Φ
233	E9	θ
234	EA	Ω
235	EB	δ
236	EC	∞
237	ED	∅
238	EE	∈
239	EF	∩
240	F0	≡
241	F1	±
242	F2	≥
243	F3	≤
244	F4	⌠
245	F5	⌡
246	F6	÷
247	F7	≈
248	F8	°
249	F9	•
250	FA	·
251	FB	√
252	FC	ⁿ
253	FD	²
254	FE	■
255	FF	(blank 'FF')

B

Device I/O Errors

If a disk or device error occurs during a command or program, MS-DOS displays an error message in the following format:

```
<yyy> error <I/O action> <device x>
Abort, Ignore, Retry:
```

In this message, < yyy > can be one of the following:

Write protect error
Bad unit error
Not ready error
Bad command error
Data error
Bad call format error
Seek error
Non-MS-DOS disk error
Sector not found error
No paper error
Write fault error
Read fault error
Disk error

The < I/O-action > can be either of the following:

READING
WRITING

< device x > indicates the device or the drive on which the error has occurred.

"Abort, Ignore, Retry:" is the prompt that MS-DOS provides. MS-DOS waits for you to enter one of the following responses:

- ▶ **A** for Abort. Abort ends the program requesting the disk read or write and you return to MS-DOS.
- ▶ **I** for Ignore. Ignore tells MS-DOS to ignore the problem and to pretend the error did not occur.
- ▶ **R** for Retry. Retry tells MS-DOS to repeat the operation. Use this response when you have corrected the error. For example, specify Retry if you have closed the drive door or removed a write-protect tab after getting a NOT READY or WRITE PROTECT error.

Usually, you should attempt recovery in this order: First specify R to try again; if this is unsuccessful then specify A to end the program. At this point, you will probably want to try a new disk or investigate the problem.

You may receive this error message which might be related to faulty disk read or write:

```
FILE ALLOCATION TABLE BAD FOR DRIVE x
```

This message means that the allocation table copy that resides in memory has pointers to nonexistent blocks. Perhaps your disk was formatted incorrectly, or was not formatted at all. Run CHKDSK and this should help you define the problem. If this error persists, you cannot use the disk. Try using a freshly formatted disk.

ANSI Escape Sequences

An ANSI escape sequence is a series of characters beginning with an escape character or keystroke that you can use to define functions to MS-DOS. With escape sequences, you can:

- ▶ Reassign keys
- ▶ Change graphics functions
- ▶ Affect cursor movement

There are two different ANSI programs for the two modes. To execute the correct version of ANSI when you are in I mode, enter the following command in the ICONFIG.SYS file, to load ANSI.SYS:

```
DEVICE = ANSI.SYS
```

When you are in V mode, you enter the following command at the system prompt:

```
ANSI
```

When you enter this command in V mode, you are actually executing ANSI.EXE, the ANSI.SYS emulator for V mode. ANSI.EXE installs itself in high memory.

ANSI.SYS and ANSI.EXE escape sequences work in the same manner. Because some of the mode and graphics rendition settings do not apply to V mode, notes have been added to the following descriptions where necessary to explain these differences. If you do specify an unrecognized attribute or mode in ANSI.EXE, it is ignored.

The default value is used when you do not specify an explicit value or when you specify zero.

Pn means numeric parameter. Pn is a decimal number specified with ASCII digits.

Ps means selective parameter. Ps is any decimal number that selects a subfunction. You can select multiple subfunctions by separating the parameters with semicolons.

C.1 Cursor Functions

The following escape sequences affect the cursor position on the screen.

CPR—Cursor Position Report (from console driver to system)

```
ESC [ Pn ; Pn R
```

The CPR sequence reports the current cursor position by way of standard input. The first parameter specifies the current line; the second parameter specifies the current column.

Note: CPR is not in ANSI.EXE.

CUB—Cursor Backward

```
ESC [ Pn D
```

This escape sequence moves the cursor back one column without changing lines. The value of Pn determines the number of columns moved. The default value for Pn is 1. The CUB sequence is ignored if the cursor is already in the far left column.

CUD—Cursor Down

ESC [Pn B

This sequence moves the cursor down one line without changing columns. The value of Pn determines the number of lines moved. The default value of Pn is 1. The CUD sequence is ignored if the cursor is already on the bottom line.

CUF—Cursor Forward

ESC [Pn C

The CUF sequence moves the cursor forward one column without changing lines. The value of Pn determines the number of columns moved. The default value for Pn is 1. The CUF sequence is ignored if the cursor is already in the far right column.

CUP—Cursor Position

ESC [Pl ; Pc H

HVP—Horizontal and Vertical Position

ESC [Pl ; Pc f

CUP and HVP move the cursor to the position you specify with parameters. The first parameter specifies the line number; the second parameter specifies the column number. The default value is 1. If you do not specify any parameters, the cursor moves to the home position.

CUU—Cursor Up

ESC [Pn A

This sequence moves the cursor up one line without changing columns. The value of Pn determines the number of lines moved. The default value for Pn is 1. The CUU sequence is ignored if the cursor is already on the top line.

DSR—Device Status Report

ESC [n

The console driver outputs a CPR sequence on receipt of the DSR escape sequence.

RCP—Restore Cursor Position

ESC [u

This sequence restores the cursor position to the value it had when the console driver received the SCP sequence (see below).

SCP—Save Cursor Position

ESC [s

The current cursor position is saved. This cursor position can be restored with the RCP sequence.

C.2 Erasing

The following escape sequences affect erase functions.

ED—Erase Display

```
ESC [ J
```

The ED sequence erases the screen, and the cursor goes to the home position.

EL—Erase Line

```
ESC [ K
```

This sequence erases from the cursor position to the end of the line.

C.3 Modes of Operation

The following escape sequences affect screen graphics.

RM—Reset Mode

```
ESC [ = Ps 1
```

or

```
ESC [ = 1
```

or

```
ESC [ = 0 1
```

or

```
ESC [ ? 7 1
```

Parameters for RM are the same as for SM (Set Mode) except that parameter 7 resets the wrap at the end-of-line mode.

Note: In V mode, when you are resetting mode parameters, only parameter 7 (no wrap at end of line) is recognized.

SGR—Set Graphics Rendition

```
ESC [ Ps ; ... ; Ps m
```

The SGR escape sequence invokes the graphics functions specified by the parameters described in Table C-1. The graphics functions remain in effect until the next occurrence of an SGR escape sequence.

Note: In V mode, only the following attributes are recognized:

0	All attributes off
1	Bold on
4	Underscore on
7	Reverse video on

Table C-1: Set Graphics Rendition Escape Sequence Parameters

PARAMETER	FUNCTION	
0	All attributes off	
1	Bold on	
4	Underscore on	(monochrome displays only)
5	Blink on	
7	Reverse Video on	
8	Concealed on	(ISO 6429 standard)
30	Black foreground	(ISO 6429 standard)
31	Red foreground	(ISO 6429 standard)
32	Green foreground	(ISO 6429 standard)
33	Yellow foreground	(ISO 6429 standard)
34	Blue foreground	(ISO 6429 standard)
35	Magenta foreground	(ISO 6429 standard)
36	Cyan foreground	(ISO 6429 standard)
37	White foreground	(ISO 6429 standard)
40	Black background	(ISO 6429 standard)
41	Red background	(ISO 6429 standard)
42	Green background	(ISO 6429 standard)
43	Yellow background	(ISO 6429 standard)
44	Blue background	(ISO 6429 standard)
45	Magenta background	(ISO 6429 standard)
46	Cyan background	(ISO 6429 standard)
47	White background	(ISO 6429 standard)

SM—Set Mode

```
ESC [ = Ps h
```

or

```
ESC [ = h
```

or

```
ESC [ = 0 h
```

or

```
ESC [ ? 7 h
```

The SM escape sequence changes the screen width or type to one of the parameters described in Table C-2.

Note: In V mode, only parameter 7 (wrap at end of line) is recognized.

Table C-2: Set Mode Escape Sequence Parameters

PARAMETER	FUNCTION
0	40 × 25 black and white
1	40 × 25 color
2	80 × 25 black and white
3	80 × 25 color
4	320 × 200 color
5	320 × 200 black and white
6	640 × 200 black and white
7	Wrap at end of line

C.4 Keyboard Reassignment

Although not part of the ANSI 3.64-1979 or ISO 6429 standard, the following keyboard reassignments are compatible with these standards. The control sequence is:

```
ESC [Pn ; ... Pn p
```

or

```
ESC ["string" ; p
```

or

```
ESC [Pn ; "string" ; Pn ; Pn ; "string" ; Pn p
```

or any other combination of strings and decimal numbers.

The final code in the control sequence (p) is one reserved for private use by the ANSI 3.64-1979 standard.

The first ASCII code in the control sequence defines which code is being mapped. The remaining numbers define the sequence of ASCII codes generated when this key is intercepted. There is one exception: if the first code in the sequence is zero (NUL), then the first and second codes make up an extended ASCII redefinition.

DEBUG

Overview

DEBUG is a debugging program that provides a controlled testing environment for binary and executable object files. DEBUG works on binary files in the same way a text editor works on source files. It lets you alter the contents of a file or CPU register, and then immediately reexecute a program to check the validity of the changes. DEBUG eliminates the need to reassemble a program to see if a problem has been fixed by a minor change.

You can abort all DEBUG commands at any time by pressing CTRL-C. CTRL-S freezes the display, so that you can read it before the output scrolls away. Pressing any other key restarts the display.

D.1 Using DEBUG

You can start DEBUG using two methods. With method 1, you type all commands in response to the DEBUG prompt. With method 2, you type all commands at the same time.

Table D-1: Methods to Start DEBUG

METHOD	COMMAND
1	DEBUG
2	DEBUG [filespec [arglist]]

D.1.1 Method 1: Prompts

To start DEBUG using method 1, type:

```
DEBUG(cr)
```

DEBUG responds with the hyphen (-) prompt, signaling that it is ready to accept your commands. Because you have not specified a filename, you can use other commands to work on current memory, disk sectors, or disk files.

Warning: When DEBUG starts, it sets up a program header at offset 0 in the program work area. On previous versions of DEBUG you could overwrite this header. You can still overwrite the default header if you do not specify a filespec. If you are debugging a .COM or .EXE file, however, do not tamper with the program header below address 5CH, or DEBUG terminates.

Do not restart a program after the "Program terminated normally" message is displayed. You must reload the program with the N and L commands for it to run correctly.

D.1.2 Method 2: Complete Command Line

To start DEBUG using a command line, type:

```
DEBUG [filespec [arglist]](cr)
```

filespec is the file to be debugged, and arglist is the rest of the command that DEBUG uses when filespec is loaded into memory. arglist is a list of filename parameters and switches that you want passed to the program filespec. You can specify an arglist if you gave a filespec. Thus, when filespec is loaded into memory, it is loaded as if it had been started with the command **filespec arglist**.

If, for example, you type:

DEBUG FILE.EXE(cr)

DEBUG loads FILE.EXE into memory starting at 100 hexadecimal in the lowest available segment. The BX:CX registers load with the number of bytes placed into memory.

D.2 Commands

Each DEBUG command consists of a single letter followed by one or more parameters. You can use any combination of upper- and lowercase letters in commands and parameters. **Note:** The control characters and the special editing functions described in this book apply here.

If you make a syntax error in a DEBUG command, DEBUG reprints the command line and indicates the error with an up-arrow (^) and the word "error." For example:

```
dcs:100 cs:110
    ^ error
```

Table D-2 summarizes DEBUG commands. They are explained in detail, with examples, later in this section.

Table D-2: DEBUG Commands

COMMAND	FUNCTION
A[address]	Assemble
C range address	Compare
D[range]	Dump
E address [list]	Enter
F range list	Fill
G[= address [address...]]	Go
H value value	Hex
I value	Input
L[address [drive record record]]	Load
M range address	Move
N filename filename	Name
O value byte	Output
Q	Quit
R[register-name]	Register
S range list	Search
T[= address][value]	Trace
U[range]	Unassemble
W[address [drive record record]]	Write

All DEBUG commands except Quit accept parameters. You can separate parameters by delimiters (spaces or commas), but you must use a delimiter between two consecutive hexadecimal values. Thus, the following commands are equivalent:

```
dcs:100 110
d cs:100 110
d,cs:100,110
```

Table D-3 defines DEBUG command parameters.

Table D-3: Command Parameters

PARAMETER	DEFINITION
drive	A one-digit hexadecimal value that indicates which drive a file is loaded from or written to. These values designate drives as follows: 0 = A:, 1 = B:, 2 = C:, 3 = D:.
byte	A two-digit hexadecimal value placed in or read from an address or register.
record	A one- to three-digit hexadecimal value that indicates the logical record number on the disk and the number of disk sectors you want to write or load. Logical records correspond to sectors; however, their numbering differs because they represent the entire disk space.
value	A hexadecimal value of up to four digits that specifies a port number or the number of times a command should repeat its functions.
address	A two-part designation consisting of either an alphabetic segment register designation or a four-digit segment address and an offset value. The segment designation or segment address can be omitted; in these cases the default segment is used. DS is the default segment for all commands except G, L, T, U, and W. For these commands, the default segment is CS. All numeric values are hexadecimal. In these addresses, for example, you must put a colon between a segment designation (whether numeric or alphabetic) and an offset: CS:0100 04BA:0100
range	range consists of two addresses, an L, and a value, where value is the number of lines the command operates on, and L80 is assumed. You cannot use the last form if another hex value follows the range, since the hex value would be interpreted as the second address of the range. For example: CS:100 110 CS:100 L 10 This example is illegal: CS:100 CS:110 ^ error The limit for range is 10000 hex. To specify a value of 10000 hex within four digits, type 0000 (or 0).

PARAMETER	DEFINITION
list	A series of byte values or strings. list must be the last parameter on the command line. For example: fcs:100 42 45 52 54 41
string	Any number of characters enclosed in quotation marks. Quotation marks can be single (') or double ("). If the delimiter quotation marks appear within a string, double the quotation marks. For example, the following strings are legal: 'This is a "string" is okay.' However, this string is illegal: 'This is a 'string' is not.' Similarly, these strings are legal: "This is a 'string' is okay." "This is a ""string"" is okay." but this string is illegal: "This is a "string" is not." Double quotation marks are not needed in the following strings: "This is a "string" is not necessary." 'This is a ""string"" is not necessary.' The ASCII values of the characters in the string are used as a list of byte values.

Assemble (A)

A[address]

Assembles 8086/8087/8088 mnemonics directly into memory.

All numeric values are hexadecimal and you must enter them as 1–4 characters. Specify prefix mnemonics in front of the opcode to which they refer. You can also enter them on a separate line.

The segment override mnemonics are CS:, DS:, ES:, and SS:. The mnemonic for the far return is RETF. String manipulation mnemonics must explicitly state the string size. For example, use MOVSW to move word strings, and MOVSB to move byte strings.

The assembler automatically assembles short, near or far jumps and calls, depending on byte displacement to the destination address. You can override the defaults with the NEAR or FAR prefix. For example:

```
0100:0500  JMP    502         ; a 2-byte short jump
0100:0502  JMP    NEAR 505    ; a 3-byte near jump
0100:505   JMP    FAR  50A    ; a 5-byte far jump
```

You can abbreviate the NEAR prefix to NE, but you cannot abbreviate the FAR prefix.

DEBUG cannot tell whether some operands refer to a word memory location or to a byte memory location. In these cases, you must explicitly state the data type with the prefix WORD PTR or BYTE PTR. You can also use WO and BY. For example:

```
NEG      BYTE PTR [128]
DEC      WO [SI]
```

DEBUG also cannot tell whether an operand refers to a memory location or to an immediate operand. DEBUG uses the common convention that operands enclosed in square brackets refer to memory. For example:

```
MOV     AX,21           ; Load AX with 21H
MOV     AX,[21]         ; Load AX with the contents
                        of memory location 21H
```

Assemble has two popular pseudo-instructions available. The DB opcode assembles byte values directly into memory. The DW opcode assembles word values directly into memory. For example:

```
DB      1,2,3,4,"THIS IS AN EXAMPLE"
DB      'THIS IS A QUOTE: "'
DB      "THIS IS A QUOTE: '"

DW      1000,2000,3000,"BACH"
```

Assemble supports all forms of the register indirect commands. For example:

```
ADD     BX,34,[BP+2].[SI-1]
POP     [BP+DI]
PUSH    [SI]
```

Assemble also supports all opcode synonyms. For example:

```
LOOPZ   100
LOOPE   100
JA      200
JNBE    200
```

For 8087 opcodes, you must explicitly specify WAIT or FWAIT. For example:

```
FWAIT FADD ST,ST(3)       ; This line will assemble an
                          ; FWAIT prefix
LD TBYTE PTR [BX]         ; This line will not
```

Compare (C)

C range address

Compares the portion of memory specified by range to a portion of the same size beginning at address.

If the two areas of memory are identical, there is no display and DEBUG returns with the MS-DOS prompt. Differences are displayed in this format:

address1 byte1 byte2 address2

These two commands have the same effect:

```
C100,200 300
C100L100 300
```

Each command compares the block of memory from 100 to 1FFH with the block of memory from 300 to 3FFH.

Dump (D)

D[range]

Displays the contents of the specified region of memory.

If you specify a range of addresses, DEBUG displays the contents. If you type the D command without parameters, 128 bytes display at the first address (DS:100) after the address displayed by the previous Dump command.

The dump displays in two portions: a hexadecimal dump (each byte is shown in hexadecimal value) and an ASCII dump (the bytes are shown in ASCII characters). Nonprinting characters are indicated by a period (.) in the ASCII portion of the display.

The display line shows 16 bytes with a hyphen between the eighth and ninth bytes. Each displayed line begins on a 16-byte boundary.

If you type the command:

```
dcs:100 110
```

DEBUG displays:

```
04BA:0100 42 45 52 54 41 ... 4E 44 TOM SAWYER
```

If you type the following command:

```
D
```

DEBUG displays 128 bytes. Each line of the display begins with an address, incremented by 16 from the address on the previous line. Each subsequent D (without parameters) displays the bytes immediately following those last displayed.

If you type the command:

```
DCS:100 L 20
```

the display is formatted as described above, but 20H bytes are displayed.

If you then type the command:

```
DCS:100 115
```

the display is formatted as described above, but all the bytes in the range from 100H to 115H in the CS segment display.

Enter (E)

E address[list]

Enters byte values into memory at the specified address.

If you type the optional list of values, DEBUG automatically replaces the byte values. If an error occurs, no byte values are changed.

If you type the address without the list, DEBUG displays the address and its contents, then repeats the address on the next line and waits for your input. At this point the Enter command waits for you to do one of the following:

1. Replace a byte value with another value by typing the value after the current value. If the value you type is not a legal hexadecimal value, or if you enter more than two digits, DEBUG does not echo the illegal or extra character.
2. Press the Space bar to advance to the next byte. To change the value, enter the new value after the current value. If you space beyond an 8-byte boundary, DEBUG starts a new display line with the address displayed at the beginning.
3. Type a hyphen (-) to return to the preceding byte. If you decide to change a byte behind the current position, type the hyphen to return the current position to the previous byte. When you type the hyphen, a new line is started with the address and its byte value displayed.
4. Press the Return key to terminate the Enter command. You can press the Return key at any byte position.

Assume you enter the following command:

```
ECS:100
```

DEBUG displays:

```
04BA:0100   EB._
```

To change this value to 41, type 41 as shown:

```
04BA:0100   EB.41_
```

To step through the subsequent bytes, press the Space bar to see:

```
04BA:0100   EB.41     10.      00.      BC._
```

To change BC to 42, type:

```
04BA:0100   EB.41     10.      00.      BC.42_
```

Now, to change 10 to 6F, type the hyphen as many times as needed to return to byte 0101 (value 10). Then replace 10 with 6F:

```
04BA:0100   EB.41     10.      00.      BC.42-
04BA:0102   00.-_
04BA:0101   10.6F_
```

Press Return to end the Enter command and return to the DEBUG command level.

Fill (F)

F range list

Fills the addresses in the range with values in the list.

If the range contains more bytes than the number of values in the list, the list is used repeatedly until all bytes in the range are filled. If the list contains more values than the number of bytes in the range, the extra values in the list are ignored. If any of the memory in the range is not valid (bad or nonexistent), the error occurs in all succeeding locations.

Assume you type the following command:

```
F04BA:100 L 100 42 45 52 54 41
```

DEBUG fills memory locations 04BA:100 through 04BA:1FF with the bytes specified. The five values are repeated until all 100H bytes are filled.

Go (G)

G[= address[address...]]

Executes the program currently in memory.

If you type only the Go command, the program runs as it would outside DEBUG.

If you set = address, execution begins at the address specified. The equals sign (=) is required, so that DEBUG can distinguish the start = address from the breakpoint addresses.

When the other optional addresses set, execution stops at the first address encountered, regardless of that address's position in the list of addresses to halt execution or program branching. When program execution reaches a breakpoint, the registers, flags, and decoded instruction display for the last instruction executed. The result is the same as if you had entered the Register command for the breakpoint address.

You can set up to ten breakpoints. Breakpoints must be set, however, only at addresses containing the first byte of an 8086-88 opcode. If you set more than ten breakpoints, DEBUG returns the BP error message. See the end of this appendix for a list of DEBUG error messages.

The user stack pointer must be valid and have 6 bytes available for this command. The Go command uses an IRET instruction to cause a jump to the program under test. The user stack pointer is set, and the user flags, Code Segment register, and Instruction Pointer are pushed on the user stack. Thus, if the user stack is not valid or is too small, MS-DOS can crash. DEBUG places an interrupt code (0CCH) at the specified breakpoint address(es).

When DEBUG encounters an instruction containing the breakpoint code, all breakpoint addresses are restored to their original intructions. If execution does not halt at one of the breakpoints, the interrupt codes are not replaced with the original instructions.

Assume you type the following command:

```
GCS:7550
```

The program currently in memory executes up to the address 7550 in the CS segment. DEBUG then displays registers and flags, and the Go command terminates.

After DEBUG encounters a breakpoint, you can type the Go command again and the program executes just as if you had typed the filename at the MS-DOS command level. The only difference is that program execution begins at the instruction after the breakpoint rather than at the usual start address.

Hex (H)

H value value

Performs hexadecimal arithmetic on the two parameters specified.

DEBUG adds the two parameters and subtracts the second parameter from the first. The results of the arithmetic display on a single line: first the sum, then the difference.

If you type the command:

```
H19F 10A
```

DEBUG performs the calculations and then displays results:

```
02A9 0095
```

Input (I)

I value

Inputs and displays one byte from the port specified by value.

This command allows a 16-bit port address.

Assume you type the following command:

```
I2F8
```

Assume also that the byte at the port is 42H. DEBUG inputs the byte and displays the value 42.

Load (L)

```
L[address [drive record record]]
```

Loads a file into memory.

Sets BX:CX to the number of bytes read. The file loaded must already be named. Name the file either when you start DEBUG or with the N command. Both the DEBUG invocation and the N command format a filename properly in the normal format of a file control block at CS:5C.

If you type the L command without any parameters, DEBUG loads the file into memory beginning at address CS:100 and sets BX:CX to the number of bytes loaded. If you type the L command with an address parameter, loading begins at the memory address specified.

If you type L with all the parameters, DEBUG loads absolute disk sectors instead of a file. The records are taken from the drive specified. The drive designation is numeric here—0 = A:, 1 = B:, 2 = C:. DEBUG begins loading with the first record specified, and continues until the number of sectors specified in the second record are loaded.

Assume you type the following commands:

```
A>DEBUG
-NFILE.COM
```

To load FILE.COM, type **L**. DEBUG loads the file and displays the DEBUG prompt. To load portions of a file or certain records from a disk, type:

```
L04BA:100 2 0F 6D
```

DEBUG then begins with logical record number 15 and loads 109 (6D hex) records into memory beginning at address 04BA:0100. When the records are loaded, DEBUG returns the hyphen prompt.

If the file has an .EXE extension, it is relocated to the load address specified in the header of the .EXE file: the address parameter is always ignored for .EXE files. The header itself is stripped off the .EXE file before it is loaded into memory. Thus, the size of an .EXE file on disk differs from its size in memory.

If the named file is a .HEX file, typing the L command with no parameters tells DEBUG to load the file beginning at the address specified in the .HEX file. If the L command includes the option address, DEBUG determines the start address by adding the address specified in the L command to the address found in the .HEX file.

Move (M)

M range address

Moves the block of memory specified by range to the location beginning at the address specified.

DEBUG always performs overlapping moves. Overlapping moves are where part of the block overlaps some of the current addresses without loss of data. Addresses that could be overwritten are moved first. When you want to move from higher addresses to lower addresses, this command moves the data beginning at the block's lowest address and then works toward the highest. When you want to move from lower addresses to higher addresses, DEBUG moves the data beginning at the block's highest address and works toward the lowest.

The M command actually copies data rather than moves it. If you do not plan to write new data to the addresses in the block you are moving, the existing data remains intact. Consequently, the sequence of the move is important.

Assume that you type:

```
MCS:100 110 CS:500
```

DEBUG first moves data from address CS:110 to address CS:510, then CS:10F to CS:50F, and so on until CS:100 is moved to CS:500. You can review the results of the move by typing the D command, with the same address you typed for the M command.

Name (N)

D

```
N filename [filename]
```

Sets file names.

The Name command performs two functions:

1. Name assigns a filename for a later Load or Write command. Thus, if you start DEBUG without naming a file to debug, you must type the N filename command before a file can be loaded.
2. Name assigns filename parameters to the file you are debugging. In this case, Name accepts a list of parameters that are used by the file being debugged.

These two functions overlap. Consider the following set of DEBUG commands:

```
-NFILE1.EXE
-L
-G
```

These commands result in four steps:

1. (N)ame assigns the filename FILE1.EXE to the filename used in any later Load or Write commands.
2. (N)ame also assigns the filename FILE1.EXE to the first filename parameter used by any program that is later debugged.
3. (L)oad loads FILE1.EXE into memory.
4. (G)o executes FILE1.EXE with FILE1.EXE as the single filename parameter, that is, FILE1.EXE is executed as if FILE1.EXE had been typed at the command level.

A more useful chain of commands might be:

```
-NFILE1.EXE
-L
-NFILE2.DAT FILE3.DAT
-G
```

Here, Name sets FILE1.EXE as the filename for the subsequent Load command. The Load command loads FILE1.EXE into memory, and then the Name command is used again, this time to specify the parameters used by FILE1.EXE. Finally, when the Go command is executed, FILE1.EXE executes as if you had typed FILE1 FILE2.DAT FILE3.DAT at the MS-DOS command level.

If DEBUG executes a Write command at this point, then FILE1.EXE—the file that is being debugged—is saved with the name FILE2.DAT. To avoid these results, always execute a Name command before either a Load or a Write command.

The Name command can affect four regions of memory:

CS:5C	FCB for file 1
CS:6C	FCB for file 2
CS:80	Count of characters
CS:81	All characters typed

DEBUG sets up a File Control Block (FCB) for the first filename parameter you gave at CS:5C. If you type a second filename parameter, you set up an FCB beginning at CS:6C. The number of characters you type, exclusive of the first character, "N", is given at location CS:80. The actual stream of characters given by the command (again, exclusive of the letter "N") begins at CS:81.

This stream of characters might contain switches and delimiters that would be legal in any command typed at the MS-DOS command level.

A typical use of the Name command is:

```
DEBUG PROG.COM
-NPARAM1 PARAM2/C
-G
-
```

In this example, the Go command executes the file in memory as if you had typed the following command line:

```
PROG PARAM1 PARAM2/C
```

Output (O)

O value byte

Send the byte specified to the output port specified by value.

This command allows a 16-bit port address.

If you type **O2F8 4F**, DEBUG outputs the byte value 4F to output port 2F8.

Quit (Q)

```
Q
```

Terminates the DEBUG utility.

The Q command takes no parameters and exits DEBUG without saving the file you are currently working on. You are returned to the MS-DOS command level.

To end the debugging session, type:

```
Q
```

DEBUG terminates and control returns to the MS-DOS command level.

Register (R)

```
R[register-name]
```

Displays the contents of one or more CPU registers.

If you do not type register-name, the R command dumps the register save area and displays the contents of all registers and flags.

If you type a register name, the 16-bit value of that register displays in hexadecimal and a colon appears as a prompt. You can then either type a value to change the register, or, if you do not want any changes, press Return.

The only valid register names are:

AX	DI	PC
BP	DX	SI
BX	ES	SP
CS	F	SS
CX	IP	SX

(IP and PC both refer to the Instruction Pointer.)

Any other entry for register-name results in a BR error message. See the end of this appendix for a list of DEBUG error messages.

If you enter F as the register-name, DEBUG displays each flag with a two-character alphabetic code. To alter any flag, type the opposite two-letter code. The flags are either set or cleared.

Table D-4 lists the flags with their codes for SET and CLEAR.

Table D-4: Register Flag Codes

FLAG NAME	SET CODE	CLEAR CODE
Overflow	OV	NV
Direction	DN (Decrement)	UP (Increment)
Interrupt	EI (Enabled)	DI (Disabled)
Sign	NG (Negative)	PL (Plus)
Zero	ZR	NZ
Auxiliary Carry	AC	NA
Parity	PE (Even)	PO (Odd)
Carry	CY	NC

Whenever you type the RF command, the flags display in a row at the beginning of a line, in the order shown in Table D-4. At the end of the list of flags, DEBUG displays a hyphen (-). You can enter new flag values as alphabetic pairs, in any order. You do not have to leave spaces between the flag entries.

To exit the R command, press the Return key. Any flags for which you did not enter new values remain unchanged.

If you enter more than one value for a flag, DEBUG returns a DF error message. If you enter a flag code other than those shown in Table D-4, DEBUG returns a BF error message. In both cases, flags up to the error in the list are changed; those flags at and after the error are not.

At startup, the segment registers are set to the bottom of free memory, the Instruction Pointer is set to 0100H, all flags are cleared, and the remaining registers are set to zero.

If you type:

R

DEBUG displays all registers, flags, and the decoded instruction for the current location. If the location is CS:11A, the display looks similar to this:

```
AX=0E00 BX=00FF CX=0007 DX=01FF SP=039D BP=0000
SI=005C DI=0000 DS=04BA ES=04BA SS=04BA CS=04BA
IP=011A  NV UP DI NG NZ AC PE NC
04BA:011A   CD21          INT 21
```

If you type:

RF

DEBUG displays the flags:

```
NV UP DI NG NZ AC PE NC - _
```

Now type any valid flag designation, in any order, with or without spaces. For example:

NV UP DI NG NZ AC PE NC - PLEICY

DEBUG responds only with the DEBUG prompt. To see the changes, type either the R or RF command:

```
RF
```

DEBUG displays:

```
NV UP EI PL NZ AC PE CY - _
```

Press Return to leave the flags this way, or to specify different flag values.

Search (S)

```
S range list
```

Searches the range specified for the list of bytes specified.

The list can contain one or more bytes, each separated by a space or comma. If the list contains more than one byte, only the first address of the byte string is returned. If the list contains only one byte, all addresses of the byte in the range are displayed.

If you type:

```
SCS:100 110 41
```

DEBUG responds with:

```
04BA:0104
04BA:010D
-type:
```

Trace (T)

T[= address][value]

Executes one instruction and displays the contents of the decoded instruction and all registers and flags.

If you type the optional = address, DEBUG traces at the address specified. The optional value tells DEBUG to execute and trace the number of steps specified by value.

The T command uses the hardware trace mode of the 8086 or 8088 microprocessor. Consequently, you can also trace instructions stored in ROM, read-only memory.

If you type:

T

DEBUG returns a display of the registers, flags, and decoded instruction for that one instruction. Assume that your current position is 04BA:011A; DEBUG might return the display:

```
AX=0E00 BX=00FF CX=0007 DX=01FF SP=039D BP=0000
SI=005C DI=0000 DS=04BA ES=04BA SS=04BA CS=04BA
IP=011A   NV UP DI NG NZ AC PE NC
04BA:011A   CD21            INT 21
```

If you type:

T=011A 10

DEBUG executes sixteen (10 hex) instructions beginning at 011A in the current segment, and then displays all registers and flags for each instruction as it is executed. The display scrolls until the last instruction is executed. Then the display stops, and you can see the register and flag values for the last few instructions performed. Remember that CTRL-S suspends the display at any time, so that you can study the registers and flags for any instruction.

Unassemble (U)

U[range]

Disassembles bytes and displays the source statements that correspond to them, with addresses and byte values.

The display of disassembled code looks like a listing for an assembled file. If you type the U command without parameters, DEBUG disassembles 20 hexadecimal bytes at the first address after that displayed by the previous Unassemble command. If you type the U command with the range parameter, then DEBUG disassembles all bytes in the range. If the range is given as an address only, then 20H bytes are disassembled instead of 80H that the Dump command would default to.

If you type:

U04BA:100 L10

DEBUG disassembles 16 bytes beginning at address 04BA:0100:

```
04BA:0100    206472   AND  [SI+72],AH
04BA:0103    69       DB   69
04BA:0104    7665     JBE  016B
04BA:0106    207370   AND  [BP+DI+70],DH
04BA:0109    65       DB   65
04BA:010A    63       DB   63
04BA:010B    69       DB   69
04BA:010C    66       DB   66
04BA:010D    69       DB   69
04BA:010E    63       DB   63
04BA:010F    61       DB   61
```

If you enter:

U04BA:0100 0108

The display shows:

```
04BA:0100    206472   AND   [SI+72],AH
04BA:0103    69       DB    69
04BA:0104    7665     JBE   016B
04BA:0106    207370   AND   [BP+DI+70],DH
```

If you change bytes in some addresses, the disassembler alters the instruction statements. You can type the U command for the changed locations, the new instructions viewed, and the disassembled code used to edit the source file.

Write (W)

W[address[drive record record]]

Writes the file being debugged to a disk file.

If you type W with no parameters, BX:CX must already be set to the number of bytes to be written; the file is written beginning from CS:100. If you type the W command with just an address, DEBUG writes the file beginning at that address.

If you use a G or T command, BX:CX must be reset before you use the Write command without parameters. Note that if DEBUG loads and modifies a file, the name, length, and starting address are all set correctly to save the modified file, as long as the length has not changed.

The file must be named either with the DEBUG invocation command or with the N command. Both the DEBUG invocation and the N command format a filename properly in the normal format of a file control block at CS:5C.

If you use the W command with parameters, DEBUG writes the file beginning from the memory address specified. The file is written to the specified drive (the drive designation is numeric here—0 = A, 1 = B, 2 = C, and so on) beginning at the logical record number specified by the first record. DEBUG continues until the number of sectors specified in the second record are written.

Note: Writing to absolute sectors is extremely dangerous because the process bypasses the file handler.

If you type:

```
W
```

DEBUG writes the file to disk and then displays the DEBUG prompt:

If you type:

```
W
CS:100 1 37 2B
```

DEBUG writes out the contents of memory, beginning with the address CS:100, to the disk in drive B. The data written out starts at logical record number 37H and consists of 2BH records. When the write is finished, DEBUG displays the prompt:

```
WCS:100 1 37 2B
-_
```

D.3 Debug Error Messages

You might see any of the following error messages during a DEBUG session. Each error terminates DEBUG command under which it occurs, but it does not terminate DEBUG itself.

ERROR CODE	DEFINITION
BF	Bad flag: You attempted to alter a flag, but the characters typed were not one of the acceptable pairs of flag values. See the Register command for the list of acceptable flag entries.
BP	Too many breakpoints: You specified more than ten breakpoints as parameters to the Go command. Retype the Go command with ten or fewer breakpoints.
BR	Bad register: You typed the R command with an invalid register name. See the Register command for the list of valid register names.
DF	Double flag: You typed two values for one flag. You can specify a flag value only once per RF command.

Index