PC

MS-DOS 3.30
Reference Manual
Volume I

Worldwide
Information
Systems

Bull

MS-DOS 3.30
REFERENCE MANUAL

SUBJECT

Description and Use of MS-DOS 3.30

ORDER NUMBER

HU95-01

January 1989

Bull

USER COMMENTS FORMS are included at the back of this manual. These forms are to be used to record any corrections, changes, or additions that will make this manual more useful.

Bull disclaims the implied warranties of merchantability and fitness for a particular purpose and makes no express warranties except as may be stated in its written agreement with and for its customer.

In no event is Bull liable to anyone for any indirect, special or consequential damages. The information and specifications in this document are subject to change without notice. Consult your Bull Marketing Representative for product or service availability.

File No.: 1513 HU95-01

PREFACE

This manual is a reference guide describing the MS-DOS operating system, Release 3.30, for the Bull family of personal computers.

Before you use this manual, you should be familiar with the information contained in your Bull PC Owner's Manual. Specifically, you should know the names and purposes of all the components of your PC, have the hardware system installed and operating, and know how to insert diskettes in diskette drives.

In addition, you should have MS-DOS 3.30 installed on your system. Installation procedures for MS-DOS are contained in the MS-DOS User's Guide (Order No. HU94).

The following resources are included on a separate diskette to assist you in working with MS-DOS:

- The Menu Manager for the level of assistance you need in typing MS-DOS commands and using application programs

- The DOS-Help Utility for help while using MS-DOS commands, explaining their proper syntax and use

Changes between previous MS-DOS software releases are summarized in this preface. The first three sections of this manual provide an overview of frequently used DOS commands and features. Detailed information on each MS-DOS command is available in Section 3.

TERMINOLOGY

In this manual, the terms "MS-DOS" and "DOS" are used interchangeably to refer to the MS-DOS operating system.

The term "diskette" in this manual is used to describe the disks that are inserted into diskette drives. The terms "hard disk" or "fixed disk" are used interchangeably to refer to disks that are not removed. The words "disk" or "disk(ette)" are used to refer to both kinds of storage media.

New diskettes are made by several manufacturers, and are called by various names, depending on the manufacturer. Diskettes that are often labeled "flexible disks" or "floppy disks" are 5-1/4 inch diskettes; 3-1/2 inch diskettes are often labeled "micro-disks" or "mini-disks." Consult the documentation shipped with your PC or accessory diskette drive for information regarding the exact kind of diskette to use. Be sure to specify that they are for use with a personal computer.

ENHANCEMENTS FOR EACH MS-DOS RELEASE

The following MS-DOS versions have been released for use with the Bull family of PCs:

- MS-DOS 3.30
- MS-DOS 3.20
- MS-DOS 3.10.10
- MS-DOS 3.10.00 (including GW-BASIC)
- MS-DOS 2.11 (including GW-BASIC)

The following sections describe the enhancements provided by each version of DOS since release 2.11.

Refer to Section 3 for complete details about MS-DOS commands for DOS 3.30.

SUMMARY OF DOS 3.30 ENHANCEMENTS (FROM DOS 3.20)

New DOS 3.30 Commands

DOS 3.30 adds the following new commands to those of release 3.20:

APPEND	to specify a search path for data files
CHCP	to support code page switching
DISPLAY.SYS	to support code page switching
EMMDRV.SYS	to emulate expanded memory in extended memory
FASTOPEN	to enhance disk performance
NLSFUNC	to support code page switching
PRINTER.SYS	to support code page switching
RAMDRIVE.SYS	to support virtual disks in expanded memory
SMARTDRV.SYS	to provide disk caching

Commands Changed In DOS 3.30

The following commands were changed or enhanced in DOS 3.30:

COUNTRY.SYS	to support code page switching
FDISK	to support multiple DOS partitions
FORMAT	to support logical drives and 1.44 MB, 3-1/2 inch floppy disks
KEYB	to support additional keyboard layouts (KEYB xx replaces KEYBxx)
MODE	to support code page switching

Most of the changes to MS-DOS are to support code page switching, and to enhance disk performance.

SUMMARY OF DOS 3.20 ENHANCEMENTS (FROM DOS 3.10.10)

New DOS 3.20 Commands

DOS 3.20 added the following new commands to those of release 3.10.10:

DRIVER.SYS	to support external disk drives
DRIVPARM	to define parameters for disk drives
ENHDRV.SYS	to support logical hard disk drives
REPLACE	to update previous versions of files
STACKS	to allocate the number and size of system stacks
XCOPY	to copy files and directories
XFDISK	to support multiple DOS partitions
XFORMAT	to format logical hard disk drives

Commands Changed In DOS 3.20

The following commands were changed or enhanced in DOS 3.20:

ATTRIB	to change the archive attribute of a file
BACKUP/ RESTORE	to backup/restore to 3-1/2 inch diskettes
COMMAND	to set environment space
COPY	to copy to/from 3-1/2 inch diskettes
COUNTRY	to allow three new country codes
DISKCOPY	to diskcopy to/from 3-1/2 inch diskettes
DISKCOMP	to compare 3-1/2 inch diskettes
FORMAT	to format 3-1/2 inch diskettes

SELECT	to allow three new country codes and support hard disks
SYS	to recognize 3-1/2 inch diskettes

Most changes to existing MS-DOS commands were to provide support for 3-1/2 inch diskettes as well as for 5-1/4 inch diskettes.

New PARK Utility In DOS 3.20

Previous versions of DOS contained a separate PARK utility for each type of Bull PC. In release 3.20, all Bull PCs use the same PARK command to protect hard disk information during transit.

SUMMARY OF MS-DOS 3.10.10 ENHANCEMENTS (FROM RELEASE 3.10.00)

The following enhancements were added by MS-DOS Release 3.10.10:

- The Menu Manager and DOS-Help Utility became available for all Bull PC models.
- All Bull PCs used a common MS-DOS release level.
- The installation process was enhanced.

SUMMARY OF MS-DOS 3.10.00 ENHANCEMENTS (FROM RELEASE 2.11)

New DOS 3.10.00 Commands

DOS 3.10.00 added the following commands to those of release 2.11:

ATTRIB	to change the read-only attribute of a file
COUNTRY	to set up international date, time, and currency formats
FCBS	to limit the number of file control blocks

JOIN	to associate two physical drives as one logical drive
LABEL	to write a volume label to a disk
LASTDRIVE	to restrict the number of disk drives accessible
SELECT	to install international message formats
SHARE	to permit a file to be shared on a network
SUBST	to permit a simple string to represent a complex path
VDISK.SYS	to simulate a virtual disk in memory

Most new commands were introduced to better support local area network applications that require file access control and protection features. Note also that VDISK replaced the DOS 2.11 RAMDISK command.

Commands Changed In DOS 3.10.00

The following commands were changed or enhanced in MS-DOS 3.10.00:

BACKUP/ RESTORE	to operate on both hard disks and diskettes, with new features to permit selective backup/restore
COPY	to handle more than 255 files
DATE/TIME	to prompt in a choice of international formats
DISKCOPY/ DISKCOMP	to handle virtual disks and high-density diskettes
FORMAT	to introduce 16-bit file allocation table pointers for more efficient space utilization on disks larger than 10 megabytes (MB)
GRAFTABL	to default to standard color graphics upon execution
GRAPHICS	to handle multiple PC-compatible graphic printers

PRINT	to include additional options for performance tuning
SORT	to treat A = a, etc., when collating ASCII characters

HIDDEN FILE NAMES BEGINNING DOS 3.10.00

Versions of DOS prior to release 3.10.00 utilized the standard Microsoft conventions for naming the hidden DOS system files IO.SYS and MSDOS.SYS. However, for additional PC compatibility, DOS 3.10.00 changed the standard hidden file names to IBMBIO.COM and IBMDOS.COM.

COMMAND PATHING FEATURE BEGINNING DOS 3.10.00

DOS 2.11 required the use of the PATH command to tell DOS in which directories to search for MS-DOS external commands. MS-DOS versions beginning with 3.10.00 can locate an external command from any directory if the drive and/or pathname of the command is specified on the command line:

```
[d:][pathname]COMMAND
```

ENHANCED DISK UTILIZATION

The allocation of disk space in DOS 3.10.00 was changed from DOS 2.11. There is a noticeable improvement in disk utilization between DOS 2.11 and versions of DOS beginning with DOS 3.10.00 on a hard disk system.

ORGANIZATION OF THIS MANUAL

Parts I through IV of this manual provide reference material to MS-DOS as follows:

PART I: WORKING WITH DOS

Describes MS-DOS concepts and conventions. If you have not previously worked with MS-DOS, be sure to read these sections, since they introduce the concepts of commands, command syntax, and file structure.

Section 1: Using MS-DOS

- Describes the purpose of an operating system, and what MS-DOS does
- Tells you how to begin using MS-DOS
- Describes how to prepare disks for use (formatting)
- Provides an overview to the use of MS-DOS.

Section 2: Learning MS-DOS

- Describes files and directories: what they are and how to use them
- Summarizes some frequently performed operations
- Describes how to copy disks and diskettes
- Discusses the use of pipes and filters to redirect command input and output
- Discusses virtual disks (memory-resident files and directories).

PART II: MS-DOS COMMANDS

Provides an alphabetical listing of MS-DOS commands. Use this section in conjunction with the DOS Help Utility to learn MS-DOS commands. This section describes the format and syntax of DOS commands in detail.

Section 3: MS-DOS Commands

- Introduces commands: types, options, entry conventions (standard formats and procedures)
- Presents the commands in reference format arranged alphabetically by name
- Describes each MS-DOS command (system, batch, and configuration)
- Describes the purpose, formats, and usage notes for each MS-DOS command
- Includes examples of command usage.

PART III: MS-DOS UTILITIES

Offers information on several DOS utility programs. Part III also contains features (batch and configuration files) that are important to improving the performance of your PC system.

Section 4: Editing, Function Keys, the Cursor, and Display Graphics

- Defines the use of the edit keys during MS-DOS command entry
- Describes the control of cursor movement
- Describes graphics display.

Section 5: Line Editor (Edlin)

- Describes the line editor, Edlin
- Describes the use of edit keys with Edlin.

Section 6: Link Program (Link)

- Describes the use of the Link utility, to produce final machine-readable versions of programs you write

Section 7: DEBUG Utility

- Describes the DEBUG utility, which provides a controlled testing environment for binary and executable object program files

Section 8: Defining System Configuration

- Describes how to use the CONFIG.SYS file and configuration commands to indicate configuration to DOS, such as supporting devices for your PC

PART IV: APPENDIXES

Appendix A: Command Format Summary Tables

- Presents a set of summary tables for DOS commands and utilities

Appendix B: Buying and Installing DOS Software

- Describes guidelines for buying, installing, and using applications software. Also includes an ASCII code chart

Appendix C: International Keyboard Considerations

- Describes international keyboard support

Appendix D: PC Support Utilities

- Describes the PC support utilities provided on the DOS diskettes

Appendix E: DOS 3.30 Master Diskette Files

- Lists the files on the DOS 3.30 Master diskettes

Appendix F: Menu Manager and DOS-Help Files

- Lists the Menu Manager, SHELUTIL, and DOS-Help files

PART V: MS-DOS MESSAGES

Appendix G: MS-DOS Messages

- Provides a complete list of messages displayed by DOS, and explains how to respond to each of them

Glossary

- Provides a list of the terminology used in this manual and in discussing MS-DOS

CONTENTS

ILLUSTRATIONS

TABLES

Section 1
USING MS-DOS

In this section:

In this section (cont):

WHAT IS AN OPERATING SYSTEM?

An operating system:

- Is a group of programs that control the functioning of the components of your PC, such as the monitor, keyboard, disk drives, and the use of system memory.

- Allows you to use other software programs, such as applications programs (word processors, spreadsheets etc.) and system development programs (programming languages and other development tools). Software programs are written for use with a specific operating system. When you buy software for your PC, specify that it is for use with MS-DOS.

- Gives you control over files. DOS uses files as a way to manage information on diskette or hard disk. (Refer to Section 2 for descriptions of files and file operations.)

  Some of the tasks that DOS can perform are:

  - Specifying to the computer the types of devices that are connected to it

  - Setting the time and date, so that the information can be saved with files

  - Preparing diskettes and hard disks for use

  - Determining the operating status of hard disks or diskettes

  - Helping you to organize files containing information or programs

  - Copying information from the memory of the PC to diskettes or hard disk

  - Copying the contents of a diskette to another diskette

  - Copying the contents of a hard disk to one or more diskettes

  - Deleting information stored on diskette or hard disk.

- Provides a link between you and your computer. The operating system communicates with you through messages and prompts. A prompt is a message displayed on the screen, indicating that the computer is waiting for you to respond before it continues.

  You communicate with the operating system by entering commands. In most cases, you enter one of the commands recognized by DOS to perform a specific task, such as the following commands for copying:

  COPY: Copies one or more files

  XCOPY: Copies files or entire directories

  REPLACE: Copies new versions of files, replacing previous version

  BACKUP: Copies files to diskettes for safekeeping

  RESTORE: Copies backup files to their original locations.

In some cases, you enter a command by pressing a a single key or selecting a choice, such as YES or NO.

The operating system and the programs that it uses are stored on diskettes or a hard disk. When you power on the PC with an operating system diskette in drive A, or with the operating system installed on the first hard disk in your system, the PC automatically loads the operating system.

THE MENU MANAGER

The Menu Manager is a utility program provided to help you learn and use DOS. The Menu Manager provides a variety of on-screen help tools and features. Instead of having to remember the exact use of a DOS command, for instance, the Menu Manager allows you to "fill-in-the-blanks." The Menu Manager provides four levels of assistance with DOS commands; the higher the level, the greater the assistance. The default level is Level 4 (the most assistance).

NOTE: The Menu Manager works with all monitors except composite monochrome graphic monitors.

Refer to the *MS-DOS User's Guide* (Order No. HU94) for information on using the Menu Manager.

SHELUTIL UTILITY

You can enable, disable, install, or delete the Menu Manager, or set up your monitor for use with the Menu Manager by using the SHELUTIL utility.

SHELUTIL also allows you to install the DOS-Help utility independent of the Menu Manager.

Refer to the *MS-DOS User's Guide* (Order No. HU94) for information on using SHELUTIL, and on installing the Menu Manager and DOS-Help files.

DOS-HELP UTILITY

The DOS-Help utility provides on-screen information about specific DOS commands. When you enter a DOS command, you can use the DOS-Help utility to get help on how to type the command, and then return to the command line without losing what you have already typed. For instance, you can check how the command name is spelled, the options that are available for the command, and other commands that can be used in conjunction with it.

Refer to the *MS-DOS User's Guide* (Order No. HU94) for more information about the DOS-Help utility.

TYPING DOS COMMANDS AND INSTRUCTIONS

You can type DOS instructions in either lowercase or uppercase. However, you must type the information in a specific sequence called the "command format." If you do not type a command in the required format, DOS either processes the command incorrectly, or cannot process the command at all.

DOS processes the information in a command line when you press the <ENTER> key to indicate that you are finished typing the command.

Depending on the keyboard you are using, the <ENTER> key is sometimes called the Return or Carriage Return key.

NOTE: Do not confuse the <ENTER> key with the <Backspace> key. The Backspace key is a dark key with a straight left-pointing arrow. The <Back Space> key erases the last character you typed.

Notational Conventions

In this manual the <ENTER> key is symbolized by a pair of angle brackets surrounding the word "ENTER." Wherever a pair of angle brackets surround a word, it indicates that you are to press the bracketed key. For example, <F1> indicates that you should press the key labeled "F1."

The same key notation is also used to indicate instances where you should press a combination of keys simultaneously, such as <Ctrl-Alt-Del>. The list of keys is separated by hyphens. <Ctrl-Alt-Del> means that you should press the Ctrl, Alt, and Delete keys at the same time.

The following key combinations are commonly used in DOS:

Purpose	Key Combination
Stop a screen from "scrolling" by; so that you can read it	<Ctrl-Num Lock> or <Ctrl-S>
Print the contents of the screen to a printer	<Shift-PrtSc>
Print data as it displays on the screen ("printer echo"). This key combination toggles the echo on or off whenever it is pressed	<Ctrl-PrtSc>
Clear the current contents from memory and reload the operating system. Also called rebooting or warm start	<Ctrl-Alt-Del>
Stop certain programs from processing further	<Ctrl-Break> or <Ctrl-C>

STARTING DOS

Depending on the model of your PC, it may contain diskette drives only, or diskette drives and hard disk(s). You may also have diskette drives of different types, or all of the same type.

Regardless of the disk system you have, you can only start DOS from an MS-DOS system diskette inserted in diskette drive A, or from hard disk drive C with DOS installed on it.

The PC searches for DOS in the following sequence:

1. It checks diskette drive A. If the drive contains a diskette, and the drive's lock lever is closed, the PC looks for DOS on the diskette.

   a. If the DOS system files are not present, the PC displays an error message and stops processing.

   b. If the operating system is present, the PC loads it into system memory, and drive A becomes the default drive.

2. If a hard disk is installed, and there is not a diskette in drive A, the PC looks for the operating system on the hard disk.

   a. If the DOS system files are not present, the PC displays an error message and stops processing.

   b. If the operating system is present, the PC loads it into system memory, and drive C becomes the default drive.

(Default drives are discussed in greater detail later in this section.)

Follow these steps to start DOS:

1. To start your PC from diskette drive A, insert an MS-DOS system diskette in drive A, and close the drive lever. (Refer to your PC Owner's manual if you need help inserting a diskette.)

   To start your PC from hard disk drive C, (with the DOS system files installed on it), do not insert a diskette, and leave the drive lever open.

2. Power on any peripheral devices attached to your PC, and then turn on your PC.

DOS and the PC perform a few diagnostics tests when you power on the system. One of the diagnostics is a check of the PC memory and the peripheral devices of your PC. The time it takes for DOS to load depends on your system configuration and memory.

The Startup Process

NOTE: This section is for technical reference. If you are a new user, you can skip to the section "Entering the Date and Time."

In order to process under DOS, the PC must find the DOS system files in the root directory of the default drive at startup. (Refer to Section 2 for details about the root directory.) The DOS system files consist of the hidden files IBMBIO.COM, IBMDOS.COM, and the command processor called COMMAND.COM (not a hidden file).

If the PC finds the DOS system files, the following processes are performed (Figure 1-1):

1. The DOS hidden system files (IBMBIO.COM and IBMDOS.COM) are executed.

2. The system configuration is loaded from battery-powered memory, and from the CONFIG.SYS file).

3. If no alternate command processor is present, COMMAND.COM is loaded.

NOTE: Alternate command processors (such as the Menu Manager) may replace the remaining steps of the startup procedure with their own.

4. If an AUTOEXEC.BAT file is present in the root directory of the default drive, it is executed.

5. If the Menu Manager is installed, the Main Menu is displayed on the screen. (Refer to the *MS-DOS User's Guide* (Order No. HU94) for details about the Menu Manager.)

6. If the Menu Manager is not installed, and there is no AUTOEXEC.BAT file, you are prompted to enter the date and time.

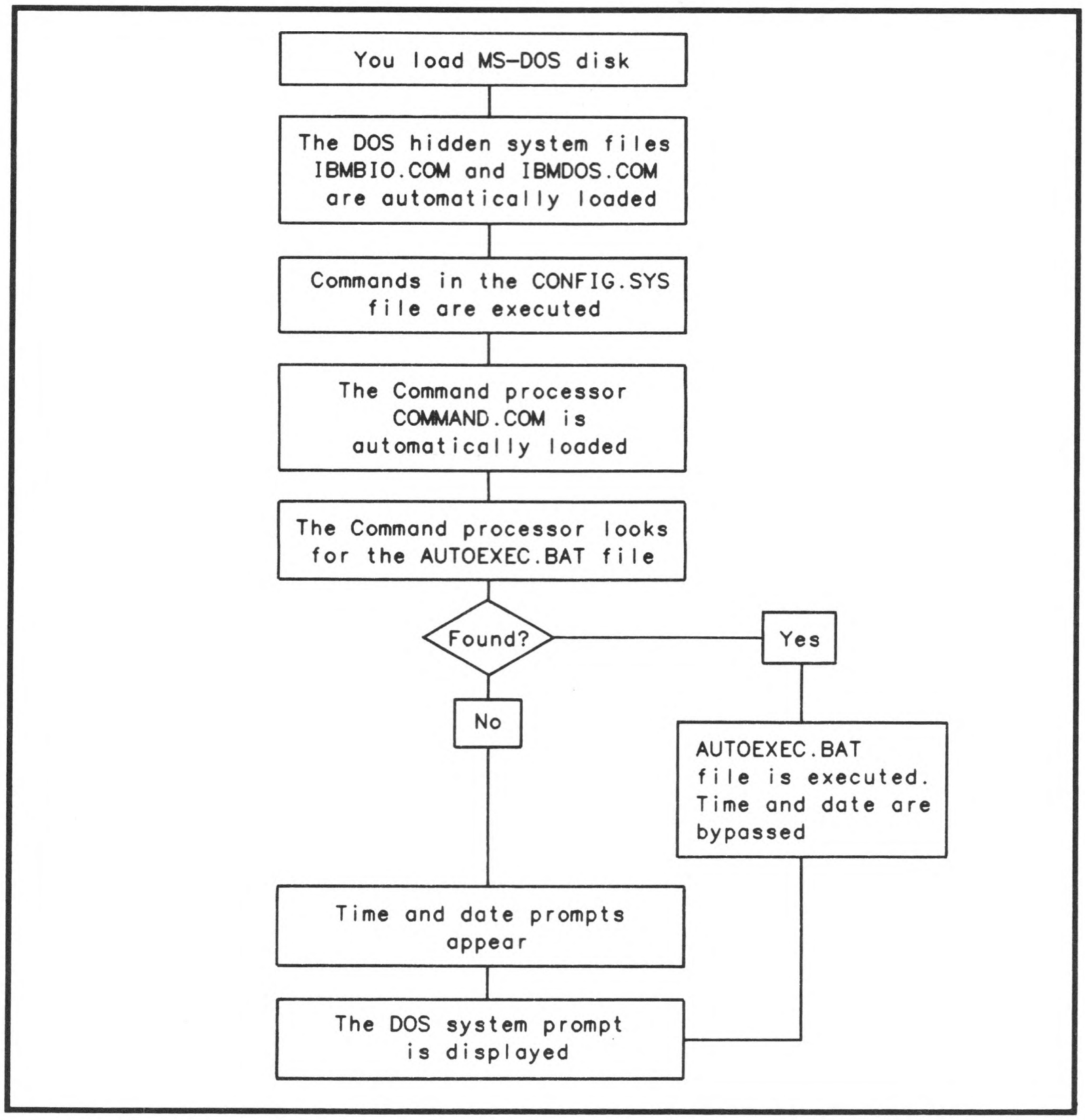


Figure 1-1. The Startup Process

Entering the Date and Time

After the diagnostic messages are displayed, the screen clears and the following message appears:

```
Bull PC
MS-DOS version 330
    (C) Copyright 1988 Bull Inc.
ROM-BIOS version X.X

Current date is Tue 1-01-1980
Enter new date (mm-dd-yy):
```

NOTE: If the Main Menu of the Menu Manager appears, refer to the *MS-DOS User's Guide* (Order No. HU94) for instructions. You can set the date and time using the Menu Manager.

Type the date in the requested format, and press the <ENTER> key. You can separate the parts of the date with a hyphen (-), a period (.), or a slash (/). For example, if the date is June 23, 1988, valid entries in U.S. format (MM-DD-YY) are:

6-23-88
6.23.88
6/23/88

NOTE: If your PC contains a battery-powered clock that stores the date and time, the date and time may already be correct. In that case, press the <ENTER> key at the date prompt and the time prompt to accept the current date and time.

If your PC is set up for international use, use the date format for your country. Refer to the COUNTRY command for details.

After you press the <ENTER> key, the screen appears as follows:

```
Bull PC
MS-DOS version 330
(C) Copyright 1988 Bull Inc.
ROM-BIOS version X.X

Current date is Tue 1-01-1980
Enter new date (mm-dd-yy): 6-23-88
Current time is  0:01:22.00
Enter new time:
```

Type in the current time, and press <ENTER>. Use a colon (:) or a period (.) to separate the hours from the minutes. Entering seconds and hundredths of a second is optional. Use continental (military) time format. For example:

One A.M. = 1:00
One P.M. = 13:00

After you have entered the date and time, the screen appears similar to the following:

```
Bull PC
MS-DOS Version 330
    (C) Copyright 1988 Bull Inc.
ROM-BIOS version X.X

Current date is Tue  1-01-1980
Enter new date (mm-dd-yy): 6-23-88
Current time is  0:01:22.00
Enter new time: 11:14

A>

A>
```

NOTE: The DOS system prompt (A>, B>, C>, etc.) appears beneath the date and time prompts. The system prompt is described later in this section.

You can also set the date and time individually at any time after startup by using the DATE and TIME commands.

NOTE: Beginning with this release of DOS, the DATE and TIME commands change the date and time stored in the battery-powered clock in AP and SP Series PCs, as well as the date and time for the current session. To permanently set other battery-powered clocks, use the software provided with the clock option.

You can change the format of the date and time display (as well as the currency symbol and the thousands-separator in numbers) by using the SELECT command. Refer to "Setting Up MS-DOS for International Use," later in this section.

THE DOS SYSTEM PROMPT

The system prompt (A>, B>, C>, etc.) indicates that DOS is ready for you to type information. The prompt is usually an alphabetic character followed by an angle bracket (>).

NOTE: If you wish, you can change the appearance of the system prompt by using the PROMPT command.

The alphabetic character indicates the current drive letter, and tells DOS where to receive and send information. DOS automatically assigns the current drive letter according to the following rules:

System Components	Drive Designations
One diskette drive	The diskette drive is designated as both A and B.
Two diskette drives	One diskette drive is designated A; the other is designated B.
One diskette drive and one hard disk	The diskette drive is designated as both A and B. The hard disk is designated C.
Two diskette drives and one hard disk	One diskette drive is designated A; the second diskette drive is B. The hard disk is C.

If you add additional disk drives to your system, they are designated as D, E, and so on.

The Default Drive

DOS needs to know the drive in which to receive and send information. The type of drive is not important. When you power on your PC, the drive where DOS is found becomes the default drive. When you type a command, DOS searches only the default drive for commands and file names you enter, and it writes files only to that drive (unless you specify a different drive). You can have DOS interact with another drive by specifying that drive in a command (as discussed in Section 2), or by changing the default drive before you type a command.

Changing the Default Drive

To change the default drive, type the letter of the drive you wish to become the new default drive, followed by a colon, and press the <ENTER> key.

To change from diskette drive A to diskette drive B, for example, type B:, and press the <ENTER> key.

The B> prompt appears. DOS now checks drive B when you type a command.

DOS COMMANDS: INTERNAL AND EXTERNAL

There are two types of DOS commands: internal and external.

Internal commands are stored within the DOS system file named COMMAND.COM, and are loaded into the memory of your PC when you power it on.

External commands are stored in separate files. DOS must be able to locate these files on disk when you use them. These commands are known as external commands because they are external to the COMMAND.COM file.

Refer to Section 3 for more information about internal and external commands.

Command-Related Messages

The following messages may appear when you type commands.

Bad command or file name

If you type a command and the following message appears,

```
Bad command or file name
```

check for a typing error. If you entered the command properly, then the command file is probably not in the default drive. If the command file is not in the default drive:

- Insert a diskette containing the file, and copy the file to the default drive. Retype the command.

- Retype the command, specifying the drive and directory in which the command file is located as part of the command.

Refer to Section 2 for details about specifying the location of external command files.

Insert COMMAND.COM

The message "Insert COMMAND.COM" indicates that you have typed an internal command, but the COMMAND.COM file is not in the default drive. Insert a diskette containing the COMMAND.COM file in the default drive (usually the MS-DOS system diskette), and press any key.

Press Any Key

When the message "Press any key" or "Strike any key when ready" appears, press any key to continue execution of a command, except those keys listed below:

- The <Ctrl>, <Num Lock>, <Alt>, <Esc>, and <Shift> keys do not transmit a character to DOS, and should not be used in this situation.
- Do not use the keys in the numeric keypad (to the right of the main keyboard).
- Do not use the cursor movement keys (labeled with arrows).
- Do not use the function keys (labeled with an F and a number).

Correcting Mistakes Within DOS

To correct a DOS command line before you press <ENTER>, press the <Back Space> key to erase one letter at a time, or, press the <Escape> key to delete the entire line, and retype it. For more information about editing DOS command lines, refer to Section 4.

If you discover you have made a mistake after pressing the <ENTER> key to start a command or program, press and hold the <Ctrl> key, then press the <Break> key. (The legend "Break" is on the front of the key labeled "Scroll Lock" in the upper right-hand corner of the keyboard.) The <Ctrl-Break> function stops the current command and redisplays the DOS prompt. You can then type another command.

PREPARING HARD DISKS AND DISKETTES FOR USE

Many of the activities of DOS involve the use of diskettes. Even if you have a hard disk system, you must use diskettes to install the hard disk or copy information to it.

New diskettes and hard disks are blank (there is no information on them). You must prepare every new diskette or hard disk for use by DOS before DOS (or programs that use DOS) can make use of it.

DOS writes and reads information in a specific fashion. It looks for certain information on a disk in specific locations. The process of preparing a disk(ette) for use by DOS is called "formatting." Formatting creates an area on the disk known as the "root directory," and prepares the disk to record information within the specifications of DOS.

You can format a new disk, or one that you want to erase and reuse, by using the FORMAT command.

NOTE: In addition to preparing a disk for use by DOS, formatting destroys the current contents of the disk.

When you use the DISKCOPY command to copy one diskette to another, the target diskette is automatically formatted as part of the copy process. Otherwise, you need to format a new disk before you can use it with DOS.

Diskettes

Write-Protect Notches And Windows

Most diskettes contain a write-protect notch (5-1/4 inch diskettes) or window (3-1/2 inch diskettes). You use this notch/window to determine whether or not information can be written onto the diskette. When the notch/window is set so that information cannot be written to the diskette, the diskette is said to be "write-protected." The method of write-protection is different for 5-1/4 inch and 3-1/2 inch diskettes.

NOTE: Some diskettes (such as the 5-1/4 inch MS-DOS Master diskettes) can only be read. You cannot write new information onto the diskette.

To write-protect a 5-1/4 inch diskette, cover the notch on the edge of the diskette (Figure 1-2) with an opaque adhesive tab (usually packaged with new diskettes). Once you have covered the notch, information cannot be written onto the diskette.

To write-protect a 3-1/2 inch diskette (Figure 1-3), switch the small plastic slider tab on the bottom of the diskette so that the write-protect window is open. Once the window is open, information cannot be written onto the diskette.

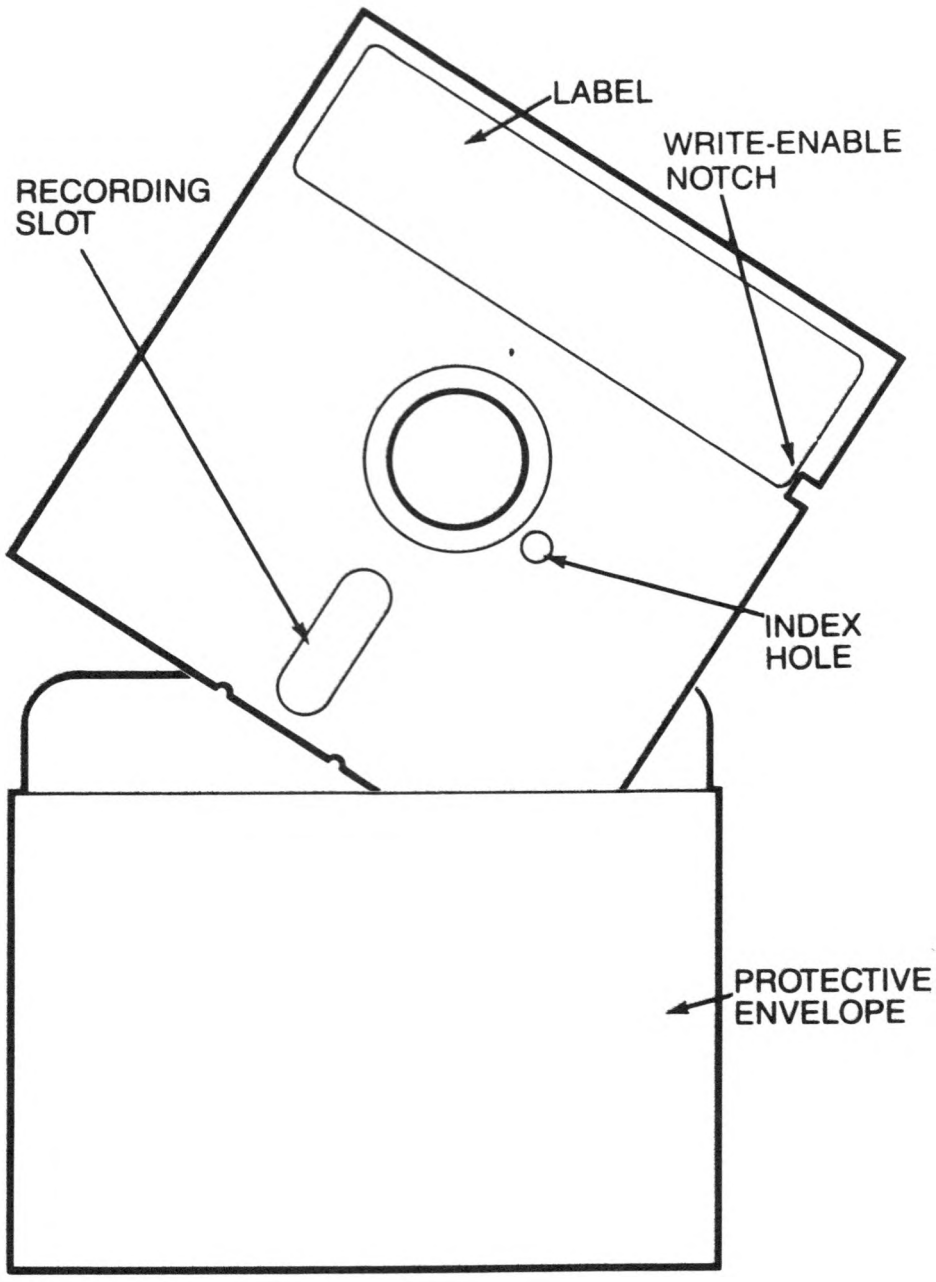


Figure 1-2. 5-1/4 Inch Diskette with Write-Protect Notch

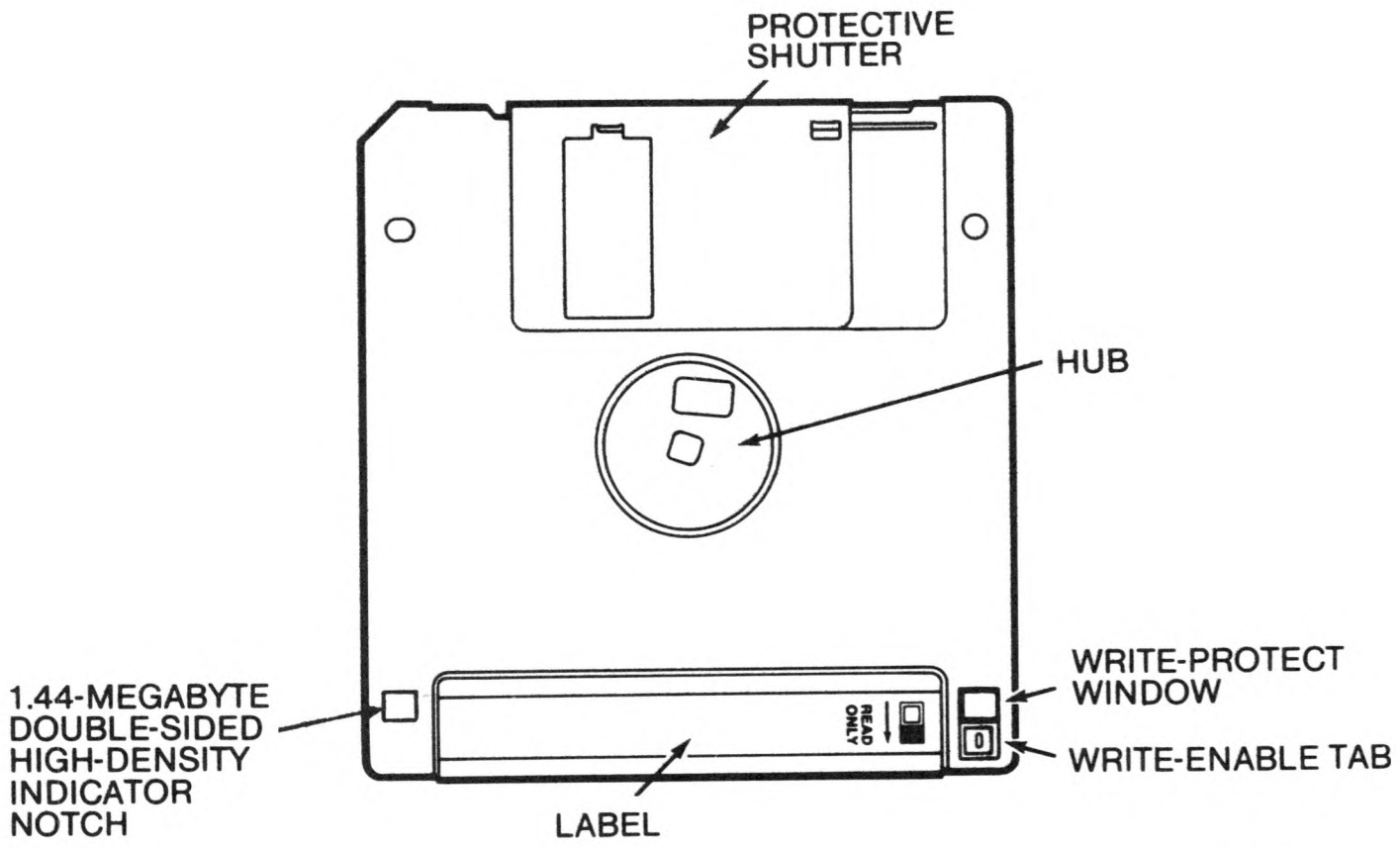


Figure 1-3. 3-1/2 Inch Diskette with Write-Protect Window

Table 1-1 shows the information storage capacity (in bytes) for several types of diskettes.

Table 1-1. Diskette Storage Capacity

Type	Sides	Tracks	Sectors/ Track	Total[a] Bytes
5-1/4 inch	1	40	8	160 KB
5-1/4 inch	1	40	9	180 KB
5-1/4 inch	2	40	8	320 KB
5-1/4 inch	2	40	9	360 KB
3-1/2 inch	2	80	9	720 KB
5-1/4 inch (High-Density)	2	80	15	1.2 MB
3-1/2 inch (High-Density)	2	80	18	1.44 MB

[a]Total bytes are usually indicated by kilobytes (KB) or by megabytes (MB). One kilobyte equals 1024 bytes. One megabyte equals 1,048,576 bytes. Therefore, 368,640 bytes is usually stated as 360 KB, and 1,258,392 bytes is usually stated as 1.2 MB.

On a diskette, each character takes 1 byte. Information is stored on "tracks" arranged in concentric circles around the center of the diskette (Figure 1-4). A track is subdivided into "sectors" of 512 bytes each. The number of tracks depends on the size of the diskette. On a diskette, track 0 is farthest from the center, and the track with the highest number is closest to the center.

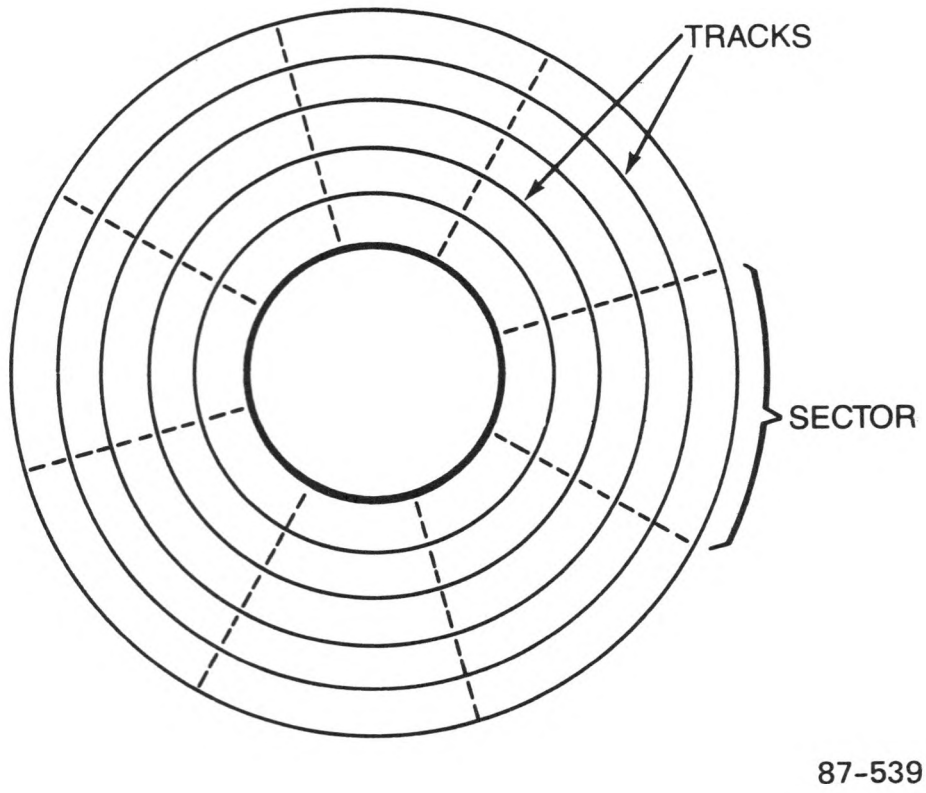


Figure 1-4. Diskette Tracks and Sectors

DOS recognizes the location of damaged bytes, sectors, and tracks and bypasses them when using a diskette.

Dividing a Hard Disk Into Partitions

If your PC contains a hard disk, you must prepare it, as you would a diskette, for use with DOS. Unlike diskettes, however, you must "partition" a hard disk before you format it.

You can have more than one operating system on a hard disk (although you can only use one operating system at a time). When you partition a hard disk, you define areas (partitions) on the disk for use by a particular operating system.

You can assign an operating system more than one partition. DOS allows you to create two partitions:

- The first DOS partition is called the "primary" partition, and can be up to 32 MB in size.

- The second partition is called the "extended" partition. You can make the extended partition any size, and can divide it into areas called "logical drives." Each logical drive can be up to 32 MB in size. DOS assigns a drive letter to each logical drive, as if it was a hard disk. For example, if you create a primary partition, and an extended partition with a single logical drive, DOS assigns the drive letter C to the primary partition, and the drive letter D to the logical drive in the extended partition. Depending on the amount of space in the extended partition, you can create logical drives up to drive letter Z.

NOTE: For information on creating partitions for other operating systems, refer to the documentation for that operating system.

To partition a hard disk, you use the FDISK command. The FDISK command creates, changes, or deletes DOS partitions. Refer to Section 3 for instructions on using the FDISK command.

IMPORTANT

If you have already installed any operating system on your hard disk, DO NOT use the FDISK command to modify partitions without a thorough understanding of the results of the partition commands.

Formatting a Hard Disk

Once you have created partitions, you must format each partition with its appropriate operating system. To format DOS partitions, you use the FORMAT command. Refer to Section 3 for instructions on using the FORMAT command.

Formatting puts a directory on the disk called the "root directory." As with diskettes, options are available during hard disk formatting to copy the system files to the hard disk and to assign a name to the hard disk. Once the hard disk is formatted, you can create files on it, or copy files to it.

IMPORTANT

Formatting destroys all of the information on a hard disk. DO NOT use the FORMAT command to reformat a hard disk without first making backup copies of all of the information you need from your hard disk.

Formatting Options

The following options are available when you format both hard disks and diskettes with the FORMAT command:

Labeling Disks

You can assign a 1- to 11-character name to the disk. This can be helpful in identifying the contents of a disk.

Formatting with System Files

If you want to use a diskette or hard disk to start DOS, you can create an MS-DOS system disk by adding the system files to the disk as part of the formatting process.

In the case of diskettes, DOS automatically attempts to format the installed diskette to the default for that specific drive. For the currently supported drive types, the defaults are as follow:

Drive Type	Tracks	Sectors/ Track	Sides
360 KB, 5-1/4 inch	40	9	2
1.2 MB, 5-1/4 inch	80	15	2
720 KB, 3-1/2 inch	80	9	2
1.44 MB, 3-1/2 inch	80	18	2

To format diskettes in a format other than the default format, or to format a diskette in a drive other than its default drive type, you must specify a formatting option or "switch" with the format command.

NOTE: Do not format diskettes to a format greater than that for which they are intended. For example, do not attempt to write a 360 KB diskette in a 1.2 MB format)

The following table lists valid formatting options:

Drive Type	Diskette Type	Format	Option
360 KB	360 KB	Single-sided	/1
360 KB	360 KB	8 sectors/track	/8
360 KB	360 KB	8 sectors/track[a]	/B
1.2 MB	360 KB	360 KB CAUTION	/4
720 KB	1.44 MB	720 KB NOT VALID	
1.44 MB	720 KB	720 KB	/N:9 /T:80
1.44 MB	1.44 MB	720 KB NOT VALID	

[a]With system space reserved.

IMPORTANT

Using single-sided or standard double-sided 5-1/4 inch diskettes formatted (or written) in a 1.2 MB drive with the /4 option (360 KB format) in a 360 KB drive can yield unpredictable results.

DO NOT format 3-1/2 inch high density (1.44 MB) diskettes in a 720 KB drive, or in the 720 KB format on a 1.44 MB drive.

Refer to Section 3 for more information about formatting disks with the FORMAT command, and specifying formatting options and codes.

GENERAL DOS OPERATIONS

The following sections provide an overview of several important DOS capabilities, and introduce a number of important DOS commands.

Displaying the Disk Directory

On a diskette or hard disk, information is stored in "files." The files are stored in "directories." To display a listing of the files in a directory, use the DIR command. Refer to Section 2 for instructions on using the DIR command with files and directories. Refer to Section 3 for a reference description of the DIR command.

Setting the Configuration of Your PC

Configuration commands define the components of your PC (such as printers, display monitors, keyboards, memory) to DOS so that it can efficiently make use of the PC resources. When you start up your PC, DOS searches for a special file called CONFIG.SYS in which you can define the configuration of your PC for DOS. Refer to Section 8 for a full discussion of configuration commands and the CONFIG.SYS file.

Processing Several Commands at Once

You can save a series of DOS commands that you use frequently in a file called a "batch" file, and perform all of the commands by simply typing the batch file name (instead of each command individually). You might, for example, have a batch file that checks the physical condition of a diskette using the CHKDSK command, then finds a file and copies it, and then compares the copy to the original. Refer to the *MS-DOS User's Guide* (Order No. HU94) for instructions on creating and using batch files.

Automatic Processing at Startup

There is a specialized batch file called AUTOEXEC.BAT. When you start or restart your PC, DOS always searches for AUTOEXEC.BAT on the system disk. This file, if present, can contain a set of commands or program names that DOS runs automatically each time you power on the system. The *MS-DOS User's Guide* (Order No. HU94) provides instructions on how to create an AUTOEXEC.BAT file.

Setting Up DOS for International Use

The date and time format delivered with DOS may not be the same as the standard format used in your country. The number delimiter and currency formats may also be different. In addition, you may need to change the standard DOS keyboard configuration to match the keyboard attached to your PC.

The SELECT command sets up an MS-DOS system diskette or your hard disk for international use. The SELECT command automatically creates a copy of the MS-DOS system diskette inserted in either drive A or drive B. SELECT uses both the KEYB command (indicating the keyboard being used) and the COUNTRY command (setting up the selected date, time, number delimiter, and currency formats). It creates an AUTOEXEC.BAT file containing the appropriate KEYB command and a CONFIG.SYS file containing the required COUNTRY command.

You can also use the KEYB and COUNTRY commands independently. Refer to the description of the individual commands in Section 3 and Appendix C for more information.

Programming the Function Keys

The keys labeled <F1> through <F10> are located in two columns on the left side of most keyboards. Some keyboards have additional function keys arranged in a row across the top. You can define (or program) function keys to perform any function you want. The programmable function keys are convenient, especially for performing frequently used or difficult commands.

For example, if you want to check the contents of a disk when you change the default drive, you can assign the DIR command to a function key. Then, to use the command, simply press the function key instead of typing the command on the keyboard.

The keys <F1> through <F6> have been predefined within DOS. For further information concerning the use of these keys, or how to alter or expand their use, refer to Section 4.

If you are using the Menu Manager, you can program function keys through the Menu Manager. Function keys <F7> through <F10> are reserved for use by the Menu Manager. Refer to the *MS-DOS User's Guide* (Order No. HU94) for further information.

Displaying Characters Not on the Keyboard

You can display a character on the screen that does not exist on your keyboard by entering its decimal code. This is accomplished by pressing the <Alt> key, and using the keys on the numeric keypad. The decimal codes for the entire character set are usually listed in programming manuals. Refer to Section 4 for further information.

Defining a Serial Printer

If you are using a printer, DOS assumes it is a parallel printer. If you are using a serial printer, you must define it to DOS with the MODE command. Your printer installation manual tells you whether the printer is parallel or serial. Some purchased application software, however, may not fully support a serial device.

PROTECTING INFORMATION

If you are processing information that cannot be replaced, or that requires a high level of security, you should take steps to ensure that data and programs are protected from accidental or unauthorized use, modification, or destruction.

Following are measures that can help maintain the integrity of the information in your files:

- Remove diskettes when not in use.
- Make frequent copies of files and diskettes.
- Write-protect diskettes.
- Back up your hard disk files to diskette regularly.
- Install equipment in a secure facility.

It is a good policy to make backup copies of all your disks. If a disk becomes damaged, or if files are accidentally erased, you still have all of the information on your backup disks. You can copy individual files with the COPY command, make a duplicate copy of an entire diskette with the DISKCOPY command, or back up an entire hard disk or directory with the BACKUP command.

With Multiple Operating Systems

You can use your PC with more than one operating system. If you plan to use DOS in addition to other operating systems, keep your data files and disk software separate. You must be sure that only compatible disk files are located on the same diskette or hard disk. Generally, data protection is simply a matter of keeping the diskettes for different operating systems properly labeled so that you always use disk files with the compatible operating system.

Remember these guidelines when using multiple operating systems:

- Keep compatible operating system software, application software, and data in the same partition of the hard disk.
- Never use a command that can destroy the contents of a disk (FORMAT, for example) without first finding out what is on the disk. If you are not sure of the exact contents, use a command to identify the contents (DIR, for example).
- Be sure to make backup copies of all important software and data.

Protecting a Hard Disk During Transit (PARK Utility)

The read/write heads of a disk drive are as sensitive as the disk surface itself. Protect the disk from being dropped or mishandled. Prior to moving a PC, you can protect the data on the hard disk by relocating the head to an unused portion of the disk using the PARK command. If you do not do this, the head could "crash" during movement of your system, and destroy information that has been recorded on the disk.

NOTE: Unlike some previous versions of MS-DOS, there is only one form of the PARK command for all Bull PC models.

Use the BACKUP command to copy the contents of your hard disk to diskette before moving the computer. Then, if the disk unit is damaged during transit, your information is safe.

Once the hard disk is backed up, follow these steps to prepare your PC for transport:

1. Type **PARK** at the DOS prompt, and press the <ENTER> key.

   If PARK command is not on the hard disk, insert the MS-DOS Options diskette in drive A, type A:PARK, and press the <ENTER> key.

2. The following screen appears:

```
                    FIXED DISK PARK UTILITY

  The PC will halt after the completion of this utility to
  position the fixed disk heads prior to movement of your PC.

Hard disk drive # 1:              Heads parked at cylinder # 615

                        S Y S T E M   H A L T E D

                        P L E A S E   P O W E R   O F F
```

3. The hard disk heads are "parked." Power off your PC.

Once you have used the PARK command, the system does not respond to any external device, including the keyboard, until the PC is restarted. If your PC is turned on, you may restart it by using a RESET button (if available), or by powering the PC off and then turning it on again.

BOOTING YOUR SYSTEM

When you make a change to the configuration of your system (or the DOS information stored in the computer memory is not current), you may have to restart your PC to begin a new session. Use the following methods:

- Turn off your PC and then turn it back on to reload DOS.
- Start the session over again by pressing <Ctrl-Alt-Del>.
- You can also use a RESET button if your PC has one.

This process is referred to as "booting" your PC. Make sure that an MS-DOS system disk is in drive A, or that DOS is installed on your hard disk. DOS is reloaded as if you had turned off your PC, and turned it back on again.

TURNING THE SYSTEM OFF

To end a DOS session, wait until current processing is completed, and the current default drive prompt (A> or C>, for example) appears. Then turn off the system and all peripheral devices. Place diskettes in protective envelopes or cases, and store them away from dust, liquid, excessive temperatures, and magnetic fields.

THE MENU MANAGER AND DOS-HELP

The Menu Manager provides an easy way to use DOS. DOS-Help provides an onscreen reference guide to DOS commands. These onscreen facilities, along with this manual and your PC owner's manual, provide the basic information you need to use your PC with DOS.

When you use your PC, you can choose to use the DOS operating system directly, or you can use it through the Menu Manager. If you are unfamiliar with DOS commands, the Menu Manager simplifies them, and provides a "fill-in-the-blanks" format.

You can also set up an applications menu, so that calling a program is a matter of choosing an item rather than typing a command name.

The Menu Manager provides a number of other useful features, such as displaying the date and time in the upper right-hand corner of the screen, and enabling you to program the function keys to perform commands you use frequently. You can leave the Main menu of the Menu Manager at any time, and return to the DOS prompt.

If you need help with an individual DOS command, you can call DOS-Help for on-screen reference, whether you are using the Menu Manager or DOS directly. The DOS-Help utility provides quick reference information on DOS commands through definitions and illustrative examples.

Refer to the *MS-DOS User's Guide* (Order No. HU94) for instructions on the installation and use of the Menu Manager and DOS-Help.

Section 2
LEARNING MS-DOS

In this section:

In this section (cont):

Summary

This section introduces some of the basic concepts of MS-DOS. It describes the use of files, including how to name them and organize them within directories and subdirectories. It also summarizes the use of many basic MS-DOS commands.

It also discusses more advanced uses of DOS, such as redirecting command input and output (so that you can direct the output of commands into files or to a printer), and how to use portions of computer memory (called virtual disks) as if they were diskettes.

WHAT ARE FILES?

A file is the smallest unit of information recognized by DOS. Files are created each time you save data, text, or program code on a disk.

A file might contain:

- Records about customers, vendors, or students
- A series of program instructions or statements that tell the PC what to do (the DOS commands are stored in files)
- A spreadsheet of monthly expenses
- A one-page memo, a letter of several pages, or a chapter of many pages for a book in a word processing package.

DOS manages files. It does not work with the contents of a file; that is left to editor programs and application programs.

WHAT ARE DIRECTORIES?

A directory is a location on a disk where you can store files. A directory can be thought of as a table of contents.

Whenever you format a disk with the FORMAT command, a single directory (called the "root" directory) is created on the disk. The root directory is the first directory DOS finds when you power on the PC.

Once a disk is formatted, you can create additional directories to store related files. A directory can also contain "subdirectories," and these, in turn, can contain further subdirectories.

The directory you are working in at any point in time is called the "default" or "working" directory, and is located on the current default disk drive. The directory contains information about the sizes of the files in the directory, their locations on the disk, and the dates that they were last changed.

THE DIRECTORY COMMAND (DIR)

To display the contents of the current directory, type:

```
DIR
```

at the DOS prompt, and press <ENTER>. As an example, the root directory on the hard disk might look like this:

```
 Volume in drive C is DOS 330
 Directory of  C:\

COMMAND  COM     24482   5-20-87    3:20a
SHELUTIL COM      3307   5-20-87    3:20a
TIME     SYS       994   5-20-87    3:20a
WTIME    COM       697   5-20-87    3:20a
HELPSCRN         108544   5-20-87    3:20a
HELP     EXE     27512   5-20-87    3:20a
HELPDIR  DCD      1023   5-20-87    3:20a
HELPDIR           2816   5-20-87    3:20a
MSDOS         <DIR>      6-23-88   12:22a
AUTOEXEC BAT        48   6-23-88   12:22a
       10 File(s)  30994432 bytes free
```

The directory listing contains the following information:

- The label assigned to the current disk and the location of the current directory are displayed at the top. (You can assign a label when you format a disk, or with the LABEL command.)

- Column 1 lists the names of files and directories contained in the current directory. In the illustration, all of the entries are files, with the exception of MSDOS, which is a directory.

- Column 2 (to the right of the file name) contains file name extensions. An extension is a part of a file name, and can be used to further identify the purpose of the file. For example, DOS commands usually have the extension .COM or .EXE, and sometimes .SYS. The full name for the AUTOEXEC file is AUTOEXEC.BAT.

- Column 3 displays the size of the file in bytes.
- Column 4 displays the creation date or the date of the last change to the file or directory.
- Column 5 displays the time that the file or directory was last written to the disk.
- The line at the bottom of the directory listing indicates the number of files in the directory, and the remaining available space (in bytes) on the entire disk.

To send a directory listing to a printer, press <Shift-PrtSc> after the directory is displayed on the screen.

Viewing a Long Directory Listing

When a directory has more files than can be displayed on the screen at one time, the first lines "scroll" off the top of the screen.

To view the directory as "pages" of information, type:

```
DIR /P
```

and press the <ENTER> key. The first page of information is displayed. To view additional pages, press any key. To bypass the rest of the pages, press <Ctrl-C>. The DOS prompt reappears when the directory listing is complete.

If the display is more than a single page long, you can send it to a printer pressing <Ctrl-PrtSc> instead of <Shift-PrtSc>. Pressing <Ctrl-PrtSc> toggles the printing on and off. Press <Ctrl-PrtSc> once to print everything that you type, or that DOS displays on the screen. Press it again to stop further printing.

To print a long directory listing:

1. Press <Ctrl-PrtSc>.
2. Type the DIR command, and press <ENTER>.
3. When the DOS prompt reappears, the directory listing is complete. Press <Ctrl-PrtSc> to turn off printing.

Another way to send the display of a long directory listing to the printer is to use a special DOS operator to redirect the output of the DIR command to the printer instead of to the monitor, as follows:

DIR > PRN or DIR >PRN

For more information on redirecting the output of a command, refer to "Command Input And Output," later in this section.

Displaying a Quick Directory Listing

If you do not need to view the sizes or dates of the files in the current directory, type:

DIR /W

and press the <ENTER> key.

This command displays file names and extensions horizontally, as shown below:

```
Volume in drive C is DOS 330
Directory of  C:\

COMMAND COM  SHELUTIL COM  TIME     SYS  WTIME   COM  HELPSCRN
HELP    EXE  HELPDIR  DCD  HELPDIR       MSDOS        AUTOEXEC BAT
       10 File(s)  30994432 bytes free
```

NAMING FILES

Information is written to disk when you assign it a file name. If the file name does not previously exist on the disk, a new file is created. If the file name matches a file name already on the disk, the new information is written over the old, destroying the previous contents of the file.

A file name and its file extension are used to identify a file. A typical DOS file name appears as follows:

```
NEWFILE.EXT
```

A file name (such as NEWFILE) can be up to eight characters long, with an extension (such as .EXT) of from one to three characters long. The extension is separated from the rest of the file name by a period (.).

You cannot use the following characters in file names or extensions:

```
. " / \ [ ] : | < > + = ; ,
```

You cannot use ASCII characters less than 20H, or include a space in a file name or in its extension.

You can use the following characters in file names and extensions:

- Any alphabet letter between A and Z
- Any digit between 0 and 9
- Any of the following special characters:

```
! @ # $ % ^ & ( ) - _ { } ' ` ~
```

You can type file names and extensions in lowercase, uppercase, or any combination of the two. DOS translates all letters into uppercase characters in the directory display.

You can also include a drive designation when you name a file. A drive designation tells DOS to look for or write the named file to the specified drive instead of the default drive. For example, the following command directs DOS to list information about the file NEWFILE.EXT, located on drive B:

```
DIR  B:NEWFILE.EXT
```

File Specifications

All of the parts of a file name comprise a *file specification*. The term file specification (or filespec) is used occasionally in this manual to indicate the complete file name format.

A file specification can be summarized as follows:

```
[d:][pathname]filename[.ext]
```

where:

[d:]

The drive designation for the disk containing the specified file. The drive designation is not required unless you need to indicate the drive on which a file is located.

[pathname]

The location of the directory that contains the file name. Pathnames may include one or more directory names. Pathnames are optional.

filename

The name of the file (up to eight characters).

[.ext]

The file name extension (up to three characters). The extension is optional.

NOTE: An entry enclosed in square brackets ([ ]) is optional.

Examples of file specifications are:

```
NEWFILE
NEWFILE.EXT
A:NEWFILE.EXT
```

The following file specifications include pathnames:

```
A:\NEWDIR\NEWFILE.EXT
A:\NEWDIR\SUBDIR\NEWFILE.EXT
```

(Pathnames are discussed later in this section.) A pathname indicates the directory location of a file, and is separated from the drive designation and file name by the backslash character (\). Each directory name is separated from the next by a backslash.

Using File Extensions

Certain file name extensions are widely used to describe the type or use of a file. Table 2-1 provides a list of some of the more commonly used extensions. Some of these extensions are automatically created by DOS, or DOS-compatible application programs. Others are in common use in computing environments.

Table 2-1. Commonly Used File Name Extensions

Extension	Indicates
ASM	Input (source coding) for Assembler programming language
BAK	Backup copy
BAS	BASIC language program (source coding)
BAT	An executable batch file
BIN	Binary image version of an executable object code file (EXE2BIN default)
COB	Compiler input (source coding) for COBOL programming language
COM	Command file on system disk
C	Compiler input (source coding) for C programming language
DAT	Common use for data files
DOC	Documentation
EXE	Any executable object code file
FOR	Compiler input (source coding) for FORTRAN programming language
LIB	Link library file input
MAC	Macro file name
MAP	Link listing default
OBJ	Object output for most programming languages (object code)
OVR	File overlay
PAS	Compiler input (source coding) for Pascal programming language
SYS	A system file
TXT	A text file
TMP	A temporary file

Reserved File Names

DOS reserves several terms for use as device names. Except as noted, the terms listed in Table 2-2 are reserved for use by DOS, and cannot be used as file names.

Table 2-2. Device Names Reserved by MS-DOS

CON	Refers to input from the keyboard or output to the screen (CONsole).
AUX	Refers to the auxiliary device attached to the first serial/parallel adapter port.
COM1	Refers to the auxiliary device attached to the first serial/parallel adapter port.
COM2	Refers to the auxiliary device attached to the second serial/parallel adapter port.
COM3	Refers to the auxiliary device attached to the third serial adapter port.
COM4	Refers to the auxiliary device attached to the fourth serial adapter port.
PRN	Refers to the first parallel printer (for output only).
LPT1	Refers to the first parallel printer (for output only).
LPT2	Refers to the second parallel printer (for output only).
LPT3	Refers to the third parallel printer (for output only).
NUL	Used when you do not want to create a particular file, but the command requires an input or output file name.

Although these names are reserved, and should not be used as file names, CON, AUX, PRN, and NUL can be used as file name extensions.

You can use these device names in place of a file name in a command to define the desired destination for data. (Refer to "Command Input and Output" in this section for more information. An example of command redirection was shown in the description of the DIR command with PRN.)

Even if you add drive designations or extensions to these reserved names, they remain associated with the devices listed in the table. For example, A:CON.XXX always refers to the console and cannot be the name of a disk file you create.

The Menu Manager, the SHELUTIL utility, and DOS-Help also make use of reserved file names. Refer to Appendix F for a list of the file names reserved for these programs.

USING WILDCARD CHARACTERS

There are times when you might want to process a group of files at once. For example, you might want to copy all of the files with the extension DOC, or delete all of the memos that begin with specific characters, such as "MAY." Rather than typing a command many times to process a group of files, you can use the question mark (?) and the asterisk (*), to indicate common elements in files, and process them all at once. When you use these characters in this manner, they are called "universal" or "wildcard" characters.

The ? Character

When you use a question mark in a file name or extension, it indicates that when the specified command is executed, any single character can occupy that position.

For example, to delete all of the memo files that begin with "MAY" from the following directory that contains memo and letter files:

```
APR12.MEM
APR12.LTR
MAY1.LTR
MAY10.MEM
MAY10.LTR
MAY15.MEM
JUN20.MEM
JUN20.LTR
JUL7.LTR
```

type:

```
DEL MAY??.MEM
```

at the DOS prompt, and press the <ENTER> key.

This command deletes all files that begin with the characters "MAY," are followed by any two characters, and end with the extension .MEM. It does not matter which two characters follow "MAY."

After the deletion, the following files remain in the directory:

```
APR12.MEM
APR12.LTR
MAY1.LTR
MAY10.LTR
JUN20.MEM
JUN20.LTR
JUL7.LTR
```

If you type:

```
DEL ???1?.???
```

and press the <ENTER> key, only the following files remain:

```
MAY1.LTR
JUN20.MEM
JUN20.LTR
JUL7.LTR
```

The * Character

Unlike the question mark, the asterisk represents any number of characters at a specified location, including no character at all.

For example, if you want to delete all of the files with the extension LTR, from the following directory:

```
APR12.MEM
APR12.LTR
MAY1.LTR
MAY10.MEM
MAY10.LTR
MAY15.MEM
JUN20.MEM
JUN20.LTR
JUL7.LTR
```

type:

```
DEL *.LTR
```

at the DOS prompt, and press the <ENTER> key. All of the files with the extension LTR are deleted, and the following files remain:

```
APR12.MEM
MAY10.MEM
MAY15.MEM
JUN20.MEM
```

If you type a character after an asterisk in a file name or extension, the characters after the asterisk are ignored. For example, in the following directory:

```
TEST
TEST.BAT
TEST1.BAT
TESTRUN.BAT
TESTRUN.BAK
TESTRAN.BAT
```

all files are deleted if you type either:

```
DEL  TEST*.*      or      DEL  TEST*N.*T
```

In the following directory, you can delete both the JUN and JUL files by typing DEL JU*.*.

```
APR12.MEM
APR12.LTR
MAY1.LTR
MAY10.MEM
MAY10.LTR
MAY15.MEM
JUN20.MEM
JUN20.LTR
JUL7.LTR
```

The following example is useful if you have given all of your text files the extension of .TXT:

```
DIR  A:*.TXT
```

Using Wildcard Characters To Process All Files

The wildcard designation *.* selects all files on the disk. It represents any file name and any file extension.

For example, if you type either of the following commands, all of the files in the current directory are deleted:

```
DEL *.*        or        ERASE *.*
```

CAUTION

The designation *.* is powerful when used with DOS commands. DEL *.* or ERASE *.* deletes every file in the current directory (except hidden files such as the two MS-DOS system files IBMBIO.COM and IBMDOS.COM). Once you erase a file from a disk, there is no way to retrieve it, except from a backup disk. When you erase the file COMMAND.COM from a system disk, for example, you cannot start DOS from that disk until you restore the COMMAND.COM file. Be very careful not to delete files that you need.

USING DIRECTORIES TO ORGANIZE FILES

The directory in which you are currently working is called the current directory, and is located on the default disk drive.

The examples that follow use a root directory that contains the following:

- The MS-DOS system files (only COMMAND.COM is visible)
- The Menu Manager files (only SHELUTIL.COM, TIME.SYS, and WTIME.COM are visible)
- The DOS-Help files (there are four HELP files)
- The subdirectory MSDOS (the MSDOS subdirectory contains all of the MS-DOS command and utility files)
- The startup batch file AUTOEXEC.BAT (contains startup instructions for the PC).

A directory listing of the example root directory, would appear as follows:

```
 Volume in drive C is DOS 330
 Directory of  C:\

COMMAND  COM     24482   6-23-88   3:20a
SHELUTIL COM      3307   6-23-88   3:20a
TIME     SYS       994   6-23-88   3:20a
WTIME    COM       697   6-23-88   3:20a
HELPSCRN        108544   6-23-88   3:20a
HELP     EXE     27512   6-23-88   3:20a
HELPDIR  DCD      1023   6-23-88   3:20a
HELPDIR           2816   6-23-88   3:20a
MSDOS         <DIR>      1-01-80  12:22a
AUTOEXEC BAT        48   1-01-80  12:22a
       10 File(s)  30994432 bytes free
```

A hard disk, or high capacity diskette, can contain files for a variety of programs and applications (such as MS-DOS program files, word processor program and document files, spreadsheets, data base, and other applications programs). If all of these files are located in the root directory, the list of files becomes very long, and it is difficult to locate the files that belong to any specific application. Also, if you store a large number of files in a single directory, it can take DOS longer to find a particular file.

To avoid the clutter and confusion of too many files in the root directory, you can move related groups of files into their own subdirectories. These subdirectories can, in turn, contain lower levels of subdirectories. A directory that contains subdirectories is called the parent directory to the subdirectories.

Creating A Subdirectory

To create subdirectories to the root directory, use the MKDIR (make directory) command. The command can be typed as MKDIR, or as MD.

For example, to create a subdirectory named WORD (to contain a word processing program), type:

MKDIR \WORD or MD \WORD

and press the <ENTER> key.

The new subdirectory is added to the root directory listing, which appears similar to the following:

```
 Volume in drive C is DOS 330
 Directory of  C:\

COMMAND  COM     24482   6-23-88   3:20a
SHELUTIL COM      3307   6-23-88   3:20a
TIME     SYS       994   6-23-88   3:20a
WTIME    COM       697   6-23-88   3:20a
HELPSCRN         108544   6-23-88   3:20a
HELP     EXE     27512   6-23-88   3:20a
HELPDIR  DCD      1023   6-23-88   3:20a
HELPDIR           2816   6-23-88   3:20a
MSDOS         <DIR>      1-01-80  12:22a
AUTOEXEC BAT        48   1-01-80  12:22a
WORD          <DIR>      6-23-88  11:38a

      10 File(s)  30994432 bytes free
```

NOTE: DOS creates the new subdirectories wherever there is room on the disk. The directories may not necessarily be listed in the order in which you typed them.

You can create any number of lower level subdirectories under a parent directory. For example, to create a subdirectory named MEMOS, under the parent directory named WORD, type:

```
MD \WORD\MEMOS
```

and press the <ENTER> key.

To create a second subdirectory called LTRS, under the WORD parent directory, Type:

```
MD \WORD\LTRS
```

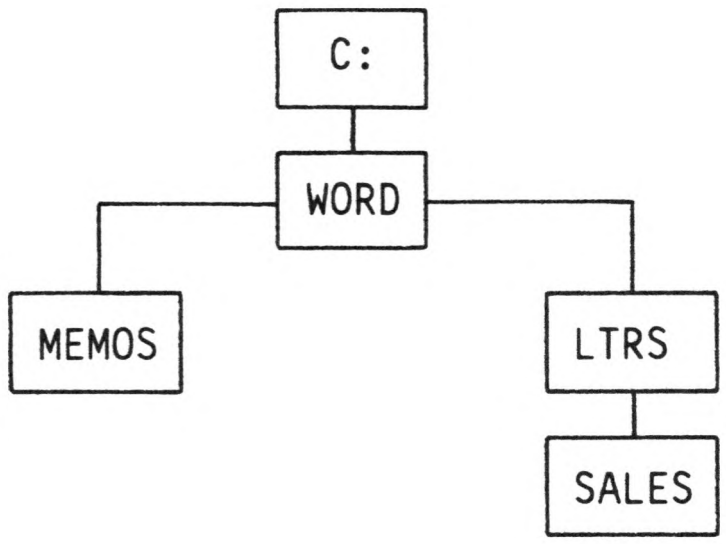
and press the <ENTER> key.

To create a still lower level subdirectory named SALES, under the LTRS subdirectory, type:

```
MD \WORD\LTRS\SALES
```

and press the <ENTER> key.

The results of the subdirectories created in these examples can be thought of as a tree, with the root at the top, as shown below:



You can list the contents of a subdirectory from within a parent directory. For example, to list the WORD subdirectory from the root directory, type:

```
DIR \WORD        or        DIR WORD
```

and press the <ENTER> key.

You can use the first of these commands (with the backslash) from any directory on the hard disk. You can only use the second command from the root directory (the parent directory) of WORD.

To list the contents of a second level subdirectory, list the parent directories in descending order, separated by backslashes. For example, to list the contents of the LTRS (second level) subdirectory from within the root directory, type:

```
DIR \WORD\LTRS
```

and press the <ENTER> key.

If a directory is empty, the directory listing appears as follows:

```
 Volume in drive C is DOS 330
 Directory of  C:\WORD\LTRS

.            <DIR>      6-23-88 11:38a
..           <DIR>      6-23-88 11:38a
       2 File(s)  30990336 bytes free
```

The two files that are indicated in the listing (. and ..), are not user files. DOS uses them to identify and manage the directory.

Uniqueness Of File Names

The same file can exist in multiple directories. For example, COMMAND.COM can be present in the root directory of the hard disk, in the MSDOS directory, and on several diskettes. A file name must be unique, however, within a single subdirectory. For example, you cannot have two COMMAND.COM files in a single directory.

Because files and subdirectories are listed together, DOS does not allow you to give the same name to a file and a subdirectory.

DISPLAYING DIRECTORY HIERARCHIES

When you have created several layers of subdirectories, you can use the TREE command to view the structure of the directories. For example, to display a directory structure for drive C, type:

```
TREE C:
```

and press the <ENTER> key.

Depending on the directories in drive C, a listing similar to the following is displayed:

```
DIRECTORY PATH LISTING FOR VOLUME DOS 330

Path: C:\MSDOS

Path: C:\WORD

Sub-directories:  MEMOS
                  LTRS

Path: C:\WORD\MEMOS

Sub-directories:  None

Path:C:\WORD\LTR

Sub-directories:  SALES

Path C:\WORD\LTR\SALES

Sub-directories:  None
```

The TREE command lists each directory on the specified disk. Subdirectories are listed with their parent directories, as well as being listed as directories in their own right.

If a printer is attached to your PC, you can print a copy of the directory structure by typing:

```
TREE C: > PRN
```

and pressing the <ENTER> key.

Substitute a different drive designation to display the structure of a different disk.

USING PATHNAMES

In order to perform a command on a file or execute a command that is located in a drive or directory other than the current directory (such as copying or deleting), you must furnish DOS with a "path" to the file, as part of the command. A pathname tells DOS the location of the file, and includes the following information:

- The drive in which the file is located
- The directory in which the file is located
- The name of the file.

For example, if the file FORMAT.COM is located in the MSDOS directory, the pathname for the file is as follows:

```
C:\MSDOS\FORMAT.COM
```

The complete pathname for a file is summarized as follows:

```
[d:][\][[subdirectory\]...\]filename[.ext]
```

where:

[d:]

The drive letter (A:, B:, C:, etc.).

[\]

The root directory of the specified disk.

[subdirectory\]

The name of the first subdirectory.

[...\]

The name of additional subdirectories.

filename

The name of the file.

[.ext]

The file extension.

NOTE: An entry enclosed in square brackets ([ ]) is optional.

You can list as many subdirectories in a pathname as you need. Use the backslash (\) to separate the subdirectory names from each other, and from the file name. The complete pathname can be up to 63 characters long.

In certain instances, it is not necessary to specify the entire pathname. For example, if you are in the MSDOS directory of drive C, and want to copy the file DEBUG.COM to a subdirectory of MSDOS, named PROGRAM, you can type the full pathnames in the command, as follows:

```
COPY  C:\MSDOS\DEBUG.COM  MSDOS\PROGRAM\DEBUG.COM
```

or, you can simplify the command, as follows:

```
COPY  DEBUG.COM  PROGRAM\DEBUG.COM
```

If you do not specify a drive or directory in a pathname, DOS searches the current drive and directory for the specified file. In the example above, because you are copying a file from the current directory (\MSDOS) to another directory in the current drive (drive C), there is no need to specify the C:\MSDOS drive and directory names.

Since you are not changing the file name when you copy it, you can simplify the command even further by typing:

```
COPY  DEBUG.COM  PROGRAM
```

and pressing the <ENTER> key.

If you accidentally delete the DEBUG.COM file from the MSDOS directory on your hard disk, you can copy the file back again from a diskette in drive A by typing:

```
COPY  A:DEBUG.COM  MSDOS\PROGRAM
```

from the root directory of drive C, and pressing the <ENTER> key. The drive designation and the initial backslash are unnecessary if your current directory is the root directory, and you are referring to the current default drive.

To copy the file from any directory on drive C, type:

```
COPY  A:DEBUG.COM  \MSDOS\PROGRAM
```

and press the <ENTER> key.

To copy the file from any drive, type:

```
COPY  A:DEBUG.COM  C:\MSDOS\PROGRAM
```

and press the <ENTER> key.

To copy all of the files from Drive A to \MSDOS, use the *.* wildcard characters, as follows:

```
COPY  A:*.*  C:\MSDOS
```

CHANGING THE CURRENT DIRECTORY

To move from one directory to another, use the Change Directory command (CHDIR or CD). Follow the command with the pathname of the directory you want to change to. For example, to make PROGRAM the new current directory, type:

```
CD  \MSDOS\PROGRAM
```

and press the <ENTER. key.

If you are in the root directory, you can also type the command as follows:

```
CD  MSDOS\PROGRAM
```

The DOS prompt reappears after you type the command. If you type the DIR command, the new current directory is displayed.

To change to the root directory, type:

```
CHDIR  \        or        CD  \
```

To move to the parent directory of the current directory, type:

```
CD ..
```

DETERMINING THE CURRENT DIRECTORY

To check your current directory, you can type the CHDIR command or CD without a pathname. The entire directory pathname is displayed, including the drive designation. For example, the following pathname appears for the PROGRAM subdirectory:

```
C:\MSDOS\PROGRAM
```

If you are currently in the root directory, the following pathname appears:

```
C:\
```

LISTING A PARENT DIRECTORY

To display the contents of a parent directory from a subdirectory, type the DIR command, followed by two periods:

```
DIR  ..
```

To display the directory listing of the next highest level parent directory, type:

```
DIR  ..\..
```

To display directories higher in the hierarchy, continue using the ..\ chain, until you reach the root directory.

DELETING A DIRECTORY

To delete a directory, use the remove directory command (RMDIR or RD). For example, to remove a directory named TEST, type:

```
RMDIR  \TEST        or        RD  \TEST
```

and press the <ENTER> key.

A directory must be empty of files (except those named "." and "..") before you can delete it. A parent directory cannot be deleted until all of its subdirectories have been deleted.

USING SUBSTITUTE PATHNAMES

The SUBST command can be used to simplify the use of pathnames. For example, rather than typing:

```
DIR  \MSDOS\PROGRAM
```

you can substitute a drive name for the pathname.

The drive name has some restrictions: it cannot be the drive name of the current drive, and it cannot be greater than the last drive letter defined to DOS. (Refer to Section 3 for more information on the LASTDRIVE and SUBST commands.)

If the last drive defined to DOS is E, you can substitute E for the pathname C:\MSDOS\PROGRAM by typing:

```
SUBST  E:  C:\MSDOS\PROGRAM
```

and pressing the <ENTER> key. Then, to display the directory of PROGRAM, type:

```
DIR  E:
```

You cannot use the substitute drive name created with the SUBST command in some DOS commands. Refer to the SUBST command description in Section 3 for more information.

USING PATHNAMES WITH COMMANDS

External Commands and Utilities

If you do not specify a directory with a command, DOS looks in the current directory for the command or utility name. External commands and utilities are independent files, not contained within COMMAND.COM. When you type an external command, the command file is read from the disk before it is executed by DOS.

If an external command is not in the current directory, you must specify its pathname in order to use the command.

For example, if the root directory is your current directory, and you want to use FORMAT.COM, which is located in the MSDOS directory, type the following:

```
C:\MSDOS\FORMAT
```

(The C:\ is optional from the root directory.)

Using the PATH Command

You can use the PATH command as an alternative to typing the pathname as part of an external command. The PATH command creates a list of directories that contain executable files. Once you include a directory in the "path" listing, you can execute the files in that directory from any drive or directory without including the pathname in the command.

For example, if the MSDOS directory is included in the path listing, rather than typing:

```
C:\MSDOS\FORMAT
```

and pressing the <ENTER> key to execute the FORMAT command (located in the MSDOS directory), you could type:

```
FORMAT
```

and press the <ENTER> key.

To place a directory in the path listing, use the following format (this example places the MSDOS directory in the path listing):

```
PATH  C:\MSDOS
```

You can include multiple directories in the path listing. For example, to include the MSDOS subdirectory, the WORD subdirectory, and the LTRS subdirectory in the path listing, type:

```
PATH  C:\MSDOS;C:\WORD;C:\WORD\LTRS
```

(Each pathname is separated from the others by a semicolon.)

DOS remembers this path listing throughout the current session (unless you type another PATH command).

Refer to Section 3 for more information on the PATH command. Refer to the *MS-DOS User's Guide* (Order No. HU94) for instructions on putting the PATH command in your AUTOEXEC.BAT file, so that you can set up the same PATH command for every session automatically.

NOTE: The PATH listing can only be used to locate files with the extensions .COM, .EXE, and .BAT. PATH does not locate data files.

Internal Commands

Internal commands do not have separate command files, but are contained within the COMMAND.COM command processor. Because they are transferred to memory as part of COMMAND.COM when you start your PC, they execute immediately after you type the command name, and press the <ENTER> key.

You can use internal commands, like external commands, to process files in drives and directories other than the current drive and directory.

- You can TYPE the contents of a file not found in the current directory by specifying its pathname as part of the file specification.

- You can copy all the files from one directory to another by typing COPY, the pathname of the source directory, and the pathname for the target directory.

- You can delete or erase all files in a directory by typing DEL or ERASE, and a directory name. Deleting the files in a directory has no effect on the files in its parent directory or in its subdirectories.

COPYING FILES, DIRECTORIES, AND DISKS

DOS provides a number of commands that transfer information from one place to another:

- The DISKCOPY command
- The COPY command
- The XCOPY command
- The REPLACE command
- The SYS command
- The SELECT command
- The BACKUP command
- The RESTORE command.

Each of these commands is summarized below. Refer to Section 3 for detailed instructions on using these commands.

Making Identical Diskette Copies (DISKCOPY)

The DISKCOPY command makes identical copies of a diskette. You can only use this command to make copies from one diskette in a diskette drive to another in the same drive, or in another drive of the same type. It cannot be used to copy from one kind of disk to another.

DISKCOPY reproduces all of the information on a diskette in the sequence encountered on the original source diskette. It does not rearrange the physical sequence of files on the disk. Like the FORMAT command, the DISKCOPY command destroys the current contents of the target diskette.

When you use the DISKCOPY command, the target diskette is formatted automatically, as part of the copy process.

Copying Individual Files (COPY)

The COPY command copies files from one place to another, regardless of the type of disk drive. Use the COPY command if you need to copy files from one kind of diskette to another, or from a diskette to a hard disk. You can copy a single file by name, a number of files, or all files in the same directory using the DOS wildcard characters * and ?. You can rename files while you are copying them, or combine two files into a third.

For example, to copy all of the files from the diskette in drive A to hard disk drive C, type the following command:

```
COPY  A:*.*  C:
```

and press the <ENTER> key.

The major restriction to using the COPY command is that all of the files you want to copy must reside in the same directory.

NOTE: If you need to copy from one diskette in a diskette drive to another diskette in the same drive, you can set up a new drive designation for the second diskette by using the DEVICE=DRIVER.SYS in your configuration file CONFIG.SYS. Refer to Sections 3 and 8 for further details.

A hard disk or diskette must be formatted before you can copy files to it.

Unlike the DISKCOPY command, the COPY command uses whatever space is available on the target disk. It does not destroy the current contents of the target disk unless the target disk has files with the same names as any of the source files. (In that case, only the files on the target diskette with the same names are replaced.)

Copying Multiple Directories (XCOPY)

Like the COPY command, the XCOPY command copies files from one place to another regardless of the type of disk drive. The XCOPY command, however, allows you to copy entire directory structures, including subdirectories.

If you want to make a copy of a 5-1/4 inch diskette (containing subdirectories) to a 3-1/2 inch diskette, you can format the new diskette in the 3-1/2 inch diskette drive and then copy all of the files to the new diskette using the XCOPY command.

There are several options that you can type in the XCOPY command line to perform a variety of copying functions. The following example uses the /S option to copy the contents of drive A (including subdirectories) into drive B:

```
XCOPY  A:  B: /S
```

The XCOPY command uses the /S, /M, and /E options to copy everything on a hard disk to diskettes:

```
XCOPY  C:  A: /S /M /E
```

The next time you use this same command, it only copies those files not previously copied. It is useful in making backup copies of files.

The XCOPY command cannot copy a file that is larger than the disk being copied to. It also cannot copy hidden files or read-protected files.

Replacing Particular Files (REPLACE)

The REPLACE command selectively replaces files on the specified diskette or hard disk. This command is useful for updating versions of MS-DOS, or newer copies of application programs.

There are several options that you can type in the REPLACE command line to perform a variety of functions. The following example uses the /S option to replace every file on the hard disk that matches a file on the diskette in drive A, no matter how many times it appears on the hard disk:

```
REPLACE  A:*.*  C: /S
```

The following example uses the /A option to add only files from the diskette in drive A that do not exist on the hard disk:

```
REPLACE  A:*.*  C: /A
```

You cannot both replace and add files with the same REPLACE command. Also, the REPLACE command cannot update hidden and system files. Like the XCOPY command, there is an option for changing the diskette in the diskette drive before the process begins, and one for prompting individually for each file.

Installing MS-DOS System Files (SYS)

The SYS command allows you to add the MS-DOS system files (IBMBIO.COM and IBMDOS.COM) to a disk so that you can use the disk to start your PC. The SYS command copies the system files to a previously formatted disk only.

The target disk must have been formatted with space reserved for the system files, or currently contain system files (i.e., from a previous DOS version), or have no files in the directory. Refer to the SYS and FORMAT command descriptions in Section 3 for further information.

The command processor COMMAND.COM is not copied to the disk with the SYS command. You must copy COMMAND.COM to the disk using a command such as COPY or XCOPY.

Creating a Custom DOS Disk (SELECT)

The SELECT command copies DOS from the diskette in either drive A or drive B, and installs it on the specified disk, allowing you to customize the disk for special keyboards or international conventions. You can use SELECT to install DOS on either a diskette or a hard disk.

You must specify the country code for the country you want (the country's telephone code), and the keyboard layout code, with the SELECT command. Refer to the descriptions of the COUNTRY and SELECT commands in Section 3 for details about the codes that are available.

The SELECT command formats the diskette (you can omit the format of a hard disk if it already contains information). The MS-DOS command files are copied to the specified directory from the original diskette. An automatic execution file (AUTOEXEC.BAT) and a configuration file (CONFIG.SYS) are created in the root directory of the specified drive. The AUTOEXEC.BAT file contains the appropriate KEYB command, and the CONFIG.SYS file contains the appropriate COUNTRY command.

Backing Up Files and Directories (BACKUP)

Use the BACKUP command to copy all information on a disk to one or more other disks for data security.

The BACKUP command protects your information. It is a good practice to back up your files regularly, to prevent loss of your data. Use the BACKUP command when you have the hard disk serviced, or if it is being shipped (especially if there is sensitive, valuable, or proprietary information on the disk).

The BACKUP command can be used to selectively copy one or more files or directories, or it can copy an entire disk. Unlike the COPY or XCOPY commands, however, the files cannot be used until they are restored with the RESTORE command. Also, unlike the XCOPY command, the BACKUP command can copy a file that is larger than a single diskette by breaking the file between diskettes. Backup files are created in sequence and cannot be restored in random order.

There are several options that you can type in the BACKUP command line to perform a variety of functions. The following command uses the /S option to back up all of the files on a hard disk to diskettes:

```
BACKUP  C:  A: /S
```

If you back up files to a hard disk, they are placed in a directory named \BACKUP. On a diskette, all files are placed in the root directory of the diskette. Any previous files in the root directory of the diskette are erased.

Options are also available to back up only files that have been modified or created since the previous backup or a specified date, to pack the files on the diskette or to create a backup log.

Since hard disks are capable of holding a great deal of information, the backup process for a hard disk usually uses many diskettes. The number of diskettes required to perform the backup depends on the amount of information on the hard disk, and the type of diskette drive you have. Have an adequate supply of formatted diskettes on hand, and number them sequentially.

Restoring Files and Directories (RESTORE)

The RESTORE command restores files previously copied to a disk with the BACKUP command. You can restore a single file, or multiple files if you use the DOS wildcard characters * and ?. An option is available to prompt you so that you can restore only the files you want from the files that match the wildcard characters.

The following command uses the /S option to restore all of the files on a set of backup diskettes back to a hard disk:

```
RESTORE  A:  C: /S
```

You are prompted to insert the diskettes in sequence.

If files in subdirectories have been backed up with the /S option of the BACKUP command, they can only be restored with the /S option of the RESTORE command. Otherwise, the RESTORE command only restores root directory files.

Terminology Used With Disk Drives

In commands, DOS uses various terms to describe the drive that contains the original information:

SOURCE
ORIGINAL
MASTER
FIRST

The drive that receives the information is described by any of the following names, even if both sets of terms refer to the same drive:

TARGET
BACKUP
NEW
DESTINATION
BLANK
SECOND

OTHER USEFUL MS-DOS COMMANDS

The following summary presents some other useful DOS commands. Refer to Section 3 for complete details about these commands, and reference descriptions of the rest of the MS-DOS commands.

Comparing Diskettes (DISKCOMP)

The DISKCOMP command compares two diskettes, track by track and sector by sector. After you have used the DISKCOPY command, DISKCOMP is useful in making sure that the diskettes are exactly alike.

The DISKCOMP command cannot be used to compare a diskette with a hard disk or another type of diskette.

DISKCOMP can only compare identical disk formats. If files have been copied into available diskette space, regardless of the location on the diskette (as with the COPY or XCOPY commands), you cannot successfully use the DISKCOMP command to compare the diskettes. Use the COMP command to compare files copied in this manner.

Comparing Files (COMP)

Use the COMP command to determine if two files contain the same information. COMP compares the contents of the files regardless of their disk locations.

Verifying Information During Copying (VERIFY)

If you specify the VERIFY ON command before you begin a copy operation, DOS verifies the information in each sector as it is written to the target disk. (You can also set verification with individual COPY or XCOPY commands.) Verification increases the time it takes for a copy operation.

Checking the Physical Condition of a Disk (CHKDSK)

The CHKDSK command reports the amount of memory used and available on a disk, including the number of files, directories, and bad sectors. An option is available for checking and correcting lost cluster errors on the disk.

Recovering Files on Damaged Disks (RECOVER)

If a file exists on a disk that has a damaged sector (whether in the directory or in the file itself), you can attempt to save it by using the RECOVER command.

COMMAND INPUT AND OUTPUT

DOS assumes that the input to a command comes from the keyboard, and the output goes to the monitor. However, this flow can be redirected. Input can come from a file rather than the keyboard, and output can go to a file or to a line printer instead of to the monitor. In addition, "pipes" can be created to allow the output of one command to become the input to another.

Redirecting Command Output

Most commands produce output that is sent to the monitor. You can send this information to a file by using a greater-than sign (>) in your command.

For example, use the DIR command without the greater-than sign to display a current directory listing:

```
DIR
```

By using the greater-than sign, as in the following example, you send the output to a file named MYFILES, instead of to the monitor:

```
DIR > MYFILES
```

If MYFILES does not already exist, DOS creates it and stores the directory listing in it. If MYFILES already exists, DOS overwrites its current contents with the new directory listing.

If you do not wish to overwrite the information in an existing target file (such as the example MYFILES), you can use two greater-than signs (>>) to append the information to the existing file.

For example, the command:

```
DIR >> MYFILES
```

appends the directory listing to a currently existing file named MYFILES. If MYFILES does not exist, it is created.

To print the current directory on a printer, use the following command:

```
DIR > PRN
```

Redirecting Command Input

It is often useful to have the input for a command come from a file, rather than from the keyboard. You can redirect command input by using a less-than sign (<) in the command. For example, the command:

```
SORT < NAMES > LIST1
```

sorts the contents of the file NAMES, and sends the output to a file named LIST1.

FILTERS

A command that reads input, transforms it and then outputs it (usually to a monitor or file), is known as a filter. Filters can be combined in many different ways, and a few filters can replace a large number of more specific commands. Filter commands can be combined with other commands to perform a "filtering" function.

The DOS filters FIND, MORE, and SORT perform the following functions:

FIND Searches for a constant string of text in a file.

MORE Displays standard monitor output one screen at a time.

SORT Arranges text in ascending or descending order from a chosen start column.

COMMAND PIPING

If you want to issue more than one command at a time, you can "pipe" commands. For example, you may need to send output from one program to another and have it serve as the input to the next program. A typical case is a program that produces output in columns and sorts the columnar output.

To pipe commands, separate them with the broken bar symbol (|). The broken bar causes all output generated on the left side of the bar to be sent to the right side of the bar for processing. For example, the command:

```
DIR | SORT
```

provides an alphabetically sorted listing of the current directory.

Piping can also be used in combination with the redirection operators (>, >>, and <). For example, the following command sorts the current directory and sends the output to a new file named DIRSORT.LIS:

```
DIR | SORT > DIRSORT.LIS
```

DOS creates a file named DIRSORT.LIS on the default drive, and redirects (>) the output from SORT to the DIRSORT.LIS file rather than to the monitor. After the command is processed, DIRSORT.LIS contains a sorted listing of the directory on the default drive (since no other drive was specified in the command). To specify a target drive other than the default drive, type:

```
DIR | SORT > B:DIRSORT.LIS
```

This command sends the sorted data to a file named DIRSORT.LIS on drive B.

A pipeline can consist of more than two filter commands. For example:

```
DIR | SORT | MORE
```

sorts the directory, then displays it one screen at a time. The message "-- MORE --" appears at the bottom of the screen when there is more output to be displayed.

SETTING UP VIRTUAL DISKS

DOS includes a feature called a virtual disk. A virtual disk allows you to store data files and directories in the computer memory, in addition to using disk storage. You can read and write these files and directories using standard DOS commands such as COPY, DIR, or TYPE as if they were on a disk.

Simulating a Disk Drive

You can use regular DOS commands to access a virtual disk, since each area set aside for storing virtual disk data files and directories is considered by DOS to be logically equivalent to an additional disk drive.

Usually a request for access to a specified drive is routed to the DOS disk-driver software residing in memory. The disk-driver, in turn, interfaces with the physical disk through the disk drive controller board. With a virtual disk, however, DOS routes requests for access to the specified drive to the virtual disk software driver routine, which interfaces directly with the virtual disk areas set aside in memory.

Retrieving or writing data stored in a virtual disk is much faster than access to a physical disk. Data transfer rates to memory and the Central Processing Unit (CPU) are faster than data transfer rates for physical disk drives. On physical disk drives, the disk transfer rate is restricted by the rotational speed of the drive. Data transfer cannot begin until the read/write head mechanism on the disk has moved to the correct track and sector within the track.

NOTE: A virtual disk area is "volatile"; each time the PC is turned off or reset (through the <Ctrl-Alt-Del> key sequence), the data in the virtual disk is lost. Therefore, make certain that virtual disk files are copied to a physical disk before turning the PC off or resetting it. Use the COPY command to copy specific files to and from the virtual disk.

Defining Virtual Disks in the Configuration File

One or more areas of computer memory must be set aside as virtual disks in order for you to use this feature. You do this by including one or more DEVICE=VDISK.SYS or DEVICE=RAMDRIVE.SYS entries in the CONFIG.SYS file on your MS-DOS system disk. (Refer to Section 8 for more information about creating a CONFIG.SYS file.)

Each time you turn on the PC or reset it using <Ctrl-Alt-Del>, DOS looks for the CONFIG.SYS file on the system disk. If the CONFIG.SYS file is present, DOS reads each command in the file, and establishes its operating conditions accordingly. A DEVICE=VDISK.SYS or DEVICE=RAMDRIVE.SYS entry in the CONFIG.SYS file causes DOS to set aside the amount of area specified in the DEVICE command for a virtual disk.

Drive Letter Assignments for Virtual Disks

DOS assigns disk drive designations each time it is started. After the standard assignments are made for physical disk drives, DOS assigns the next available drive letter to logical drives. A logical drive might be a second DOS partition on a disk or a virtual disk.

Following are examples of how to make drive letter assignments for virtual disks:

- Your PC has two diskette drives. You create a CONFIG.SYS file, and add two DEVICE=VDISK.SYS or DEVICE=RAMDRIVE.SYS entries:

  ```
  Diskette 1 and 2      = A and B
  Virtual disks 1 and 2 = C and D.
  ```

- Your PC has one diskette drive and one hard disk drive with a single DOS partition. You add a DEVICE=VDISK.SYS or DEVICE=RAMDRIVE.SYS command to an existing CONFIG.SYS file that previously contained no virtual disk entries:

  ```
  Diskette 1            = A and B
  Hard disk             = C
  Vitrual disk          = D
  ```

- Your PC has one diskette drive and one hard disk drive greater than 32 MB with two DOS partitions. You add a DEVICE=ENHDRV.SYS and a DEVICE=VDISK.SYS or DEVICE=RAMDRIVE.SYS command to your CONFIG.SYS file:

  ```
  Diskette drive        = A and B
  Partitions 1 and 2    = C and D
  Virtual disk          = E
  ```

  These designations are made each time you start DOS, until you change or delete the CONFIG.SYS file, or start DOS from a different disk.

Section 3
MS–DOS COMMANDS

In this section:

In this section (cont):

In this section (cont):

Commands are the way in which you communicate with the computer. By entering DOS commands on your keyboard, you can direct the system to perform useful tasks:

- Compare, copy, display, delete, and rename files
- Copy, format, and label disks
- Analyze and list directories
- Enter date and time
- Set various printer and screen options
- Copy DOS system files to another disk
- Use DOS utilities such as EDLIN, the DOS line editor
- Execute applications programs or your own programs

TYPES OF DOS COMMANDS

DOS has two kinds of commands: internal and external. Internal commands are part of the command processor COMMAND.COM. Once you have started your PC, DOS uses the internal commands from the memory of the PC. External commands have their own separate files. In order to use external commands, DOS must be able to locate the file.

Command Processors

The file COMMAND.COM contains the standard DOS command processor, which executes the commands you issue. You can also create your own command processor or purchase a substitute. The command processor in COMMAND.COM, or its equivalent, must be resident at system startup time in order for DOS to start up or "boot." If a command processor is not present on the default drive at system startup, the system displays the message "Bad or missing command interpreter."

Internal Commands

In addition to being a command processor, COMMAND.COM contains many basic DOS commands. These files are loaded into system memory at startup, and are available for immediate execution as long as the COMMAND.COM file is resident in memory. The only time you have to reload COMMAND.COM during a session is when another program has used the same memory. In that case, a message is displayed requesting you to insert a diskette containing COMMAND.COM.

Since internal commands are not separate files, they do not display when you list a DOS directory. COMMAND.COM is listed in the root directory.

Internal commands execute immediately after you type the command line and press the <ENTER> key.

The following internal commands are described in this section:

BREAK	IF**
BUFFERS*	LASTDRIVE*
CALL**	MKDIR (MD)
CHDIR (CD)	NLSFUNC
CLS	PATH
COMMAND	PAUSE**
COPY	PROMPT
COUNTRY*	REM**
CTTY	RENAME (REN)
DATE	RMDIR (RD)
DEL (ERASE)	SET
DEVICE*	SHELL*
DIR	SHIFT**
ECHO**	STACKS*
ERASE (DEL)	TIME
EXIT	TYPE
FCBS*	VER
FILES*	VERIFY
FOR	VOL
GOTO**	

Several commands have alternate spellings (for example, either MKDIR or MD creates a new directory). Where alternate spellings can be used, the second spelling is also provided and enclosed in parentheses.

*Configuration command.
**Batch command.

External Commands

External commands reside on disk as program files, and can be listed with the DIR command. Unlike internal commands, external commands must be read from disk before they can be executed. If the disk containing the command is not in the default directory or in the indicated directory, DOS cannot locate or execute the command. The message "Bad command or file name" is usually displayed. (The PATH command, described in this section, allows you to search prespecified directories for external commands.)

In addition to DOS commands, any file with the file extension .COM, .EXE, or .BAT is considered to be an external command. These include programs you create yourself or ones you obtain for use with DOS. When you enter the name of an external command, you do not need to include its file extension. The DOS command CHKDSK.COM, for example, is an external command that can be typed as CHKDSK.

The programs that you create with most languages (including Assembly language) have the extension .EXE (executable files). You can also use the EXE2BIN command to convert .EXE files to .COM files.

The following external commands are described in this section:

APPEND	GRAPHICS
ASSIGN	JOIN
ATTRIB	KEYB
BACKUP	LABEL
CHCP	MODE
CHKDSK	MORE
COMP	PRINT
DISKCOMP	RECOVER
DISKCOPY	REPLACE
ENHKEY	RESTORE
EXE2BIN	SELECT
FASTOPEN	SHARE
FDISK	SORT
FIND	SUBST
FORMAT	SYS
TREE	GRAFTABL
XCOPY	

COMMAND FORMAT

The command format (or command syntax) is the precise sequence and structure in which you must type a command. If you do not type commands in the proper format, DOS responds with an appropriate message. In some instances, a mistyped command can cause DOS to do something other than what you intended a command to do.

Format

The basic format for the command line of an external DOS command is:

<command PATH> <command NAME> <command OPTIONS>

More specific notation for the command path and command name portion of the command line is:

[d:][\][directory[\directory[\...]]\]filename[.ext]

This format is described in detail in the following sections.

Command Path

The command path tells DOS where to find the command file. The pathname can consist of the drive in which the command file is located, and the directory in which the command is located (preceded by all higher level directories, if any)

You use the pathname only with external commands, and its use is optional. If, for example, you type a command that is located in the current directory, you do not need to enter a path. You must, however, provide DOS with a path in order for it to locate a command file saved in another drive or directory. For example, you need to specify a pathname when a command is located in a subdirectory, and you are in a parent directory.

A command pathname consists of the following items:

[d:]	The disk drive designation [optional]
[\]	The root directory [optional]
[directory]	The first-level directory within the root directory [optional]
[\directory[\...]]	One or more additional subdirectories [optional]
[\]	The end of the command pathname part of the file specification [optional when no file name follows]

An example of a pathname for the CHKDSK command might be:

```
C:\MSDOS\CHKDSK
```

(Refer to Section 2 for a complete discussion of the use of paths and pathnames.)

Command Name

The command name in the command format is the name of the command you would like to perform, such as COPY, PRINT, or DIR.

The format for a command name is:

```
filename[.ext]
```

where the file name is a one- to eight-character name for the command, and [.ext] is an optional one- to three-character file name extension.

For example, you can type the command name CHKDSK or CHKDSK.COM.

Command Options

Many commands can be used with a variety of options. For example, the FORMAT command can be used to format either hard disks or diskettes. Since these are different types of drives, some options can be used only with diskettes and not with hard disks. For example, the following FORMAT command option formats a 360 KB diskette so that it appears to be single-sided:

```
FORMAT A: /1
```

Command Delimiters

The parts of a command must be separated by a character such as a space or comma. Characters used in this manner are known as delimiters.

Delimiters that you can use in DOS command lines include:

space	[ ]
comma	,
semicolon	;
equal sign	=
tab	→|

DOS treats multiple spaces and single spaces between parts of a command in the same way.

In the following example, a space separates the command name COPY from the file name MYFILE.OLD, and the file name MYFILE.OLD from the file name MYFILE.NEW:

```
COPY MYFILE.OLD MYFILE.NEW
```

In the following example, spaces separate the command RENAME from the first file name, and a comma separates the two file names:

```
RENAME  THISFILE,THATFILE
```

Notation Conventions

This manual uses several conventions to indicate how you should type DOS commands. The following conventions are used to indicate command formats in this manual:

CAPS — Command names and options are shown in this manual in capital letters. Although you must enter the commands and options exactly as they appear in the command format, you can type them in either uppercase or lowercase. For example, GRAFTABL is the exact spelling of a command name that you can also type as GRAFTABL.

Bold — Text that appears in boldface type represents menus, messages, or other text that appears on your screen.

lowercase — Command paths and options where you must supply a value of the appropriate type are displayed in lowercase. For example, in the following command, you would supply a file name for "filename" and a drive letter for "d:."

```
COPY filename d:
```

NOTE: In some instances in the text, options are surrounded by angle brackets, as in <filename>.

[ ] — Square brackets surround optional entries. For example, [.ext] indicates that the file extension is not a required part of the command.

| A solid vertical bar indicates that you may choose between the items on either side of the bar. For example, ON | OFF indicates that you can choose ON or OFF, but not both. The vertical bar is not a keyboard character, and therefore is never entered as part of a command from the keyboard.

NOTE: The vertical bar should not be confused with the broken bar (¦). The broken bar is a keyboard character, and when entered on the command line with a DOS filter command (FIND, MORE, and SORT) indicates a "pipe." (Piping is explained in Section 2.)

... Ellipses indicate that multiple entries of the same type can be supplied for a command element. For example, [[filename[.ext]...] indicates that you may specify more than one file for processing.

<> Angle brackets indicate specific keys on the keyboard, or a combination of keys. For example, <Ins> or <F3> represent single keystrokes. <Ctrl-Z> indicates that you should press and hold the <Ctrl> key while you press Z. <Ctrl-Break> indicates that you should press the <Ctrl> key and the <Break> keys simultaneously.

On occasion, options are indicated by angle brackets, as in <filename>. There are two special cases using angle brackets:

1. The notation <ENTER> is used to represent the Enter key, also referred to as the Return key or the Carriage Return key. This key is often labeled on the keyboard with only an arrow pointing down and to the left.

2. There is no universally accepted symbol for a space (blank) character. The notation <space> is used in some instances where a space character is critical to the format.

,;:/= All other punctuation, such as commas, semicolons, colons, slash marks, and equal signs, must be entered exactly as indicated.

Common Command Entry Conventions

The following summary applies to DOS commands:

- The DOS system prompt is represented by the default drive letter, followed by a greater-than (>) sign (unless you change the prompt using the PROMPT command). For example, the prompt for drive A is: A>. You type DOS commands at the DOS prompt.

- Disk drives are either source drives or destination drives. A source drive is the drive from which you transfer information; a destination (target) drive is the drive that receives the information from the source.

- You can precede external commands with a drive designation and/or a pathname to indicate in which drive and directory DOS can find the command file.

- You must include the file extension when referring to a command with a file extension other than .EXE, .COM, and .BAT.

- Command names are usually followed by one or more options.

- You can type command names and options in either uppercase or lowercase, or any combination of the two.

- In many commands, you can use the wildcard characters * and ? to indicate characters in file names or pathnames you specify. (Refer to Section 2 for details.)

- You cannot use reserved device names (for example, PRN, LPT1, AUX, NUL, COM1, or CON) in commands and file names you create. (Refer to Section 2 for details about reserved device names.)

- In this manual, the <space> character is used as the delimiter between the parts of a command.

- The period is a delimiter for file extensions. Do not use any other delimiter.

- You can use the editing and function keys when you type commands. (Refer to Section 4 for a description of these keys.)

- Commands take effect only after you have pressed the <ENTER> key.

- You can stop commands when they are running, by pressing <Ctrl-Break> or <Ctrl-C>.

- When instructions say "Strike any key" or "Press any key" or "-- More --", you can press any alphabetic key (A to Z), any numeric key (0 to 9), the space bar, or the <ENTER> key.

- When commands produce a large amount of output on the screen, the display automatically scrolls by. To temporarily stop the scrolling, you can press <Ctrl-Num Lock> or <Ctrl-S>. Press any key to resume scrolling.

SUMMARY OF COMMANDS

Appendix A provides an alphabetical listing of each DOS command, briefly explaining its purpose and showing its format.

COMMAND DESCRIPTIONS

The following pages contain detailed descriptions of each of the DOS commands. The commands are described in alphabetical order.

ANSI.SYS

TYPE: Configuration

Refer to the DEVICE command description.

APPEND

APPEND

TYPE: Internal External

PURPOSE: Creates a path listing for files that have extensions other than .COM, .EXE, and .BAT. The path allows DOS to locate the files outside of the current directory.

FORMAT: The first time you load APPEND:

[d:][pathname]APPEND d:pathname[;[d:]pathname ...]

or

[d:][pathname]APPEND [/X] [/E]

After you have loaded APPEND:

(to display the current append paths):

[d:][pathname]APPEND

or

(to replace the current APPEND paths):

[d:][pathname]APPEND d:pathname[;[d:]pathname ...]

WHERE: [d:] [pathname]

(*Before* the APPEND command) Indicates the drive and path that contain the APPEND command file.

[d:] [pathname]

(*After* the APPEND command) Specifies the drive and path to search. Paths cannot be specified the first time the APPEND command is loaded if either /X or /E are also specified.

;

Separates the APPEND paths. Cancels the current APPEND paths when used as the only parameter, as shown below:

APPEND ;

/X

Processes SEARCH FIRST, FIND FIRST, and EXEC functions.

Commands such as DIR and COMP use the SEARCH FIRST function to search for files. Commands such as BACKUP, TREE, and RESTORE use the FIND FIRST function to locate files. DOS uses the EXEC function any time a command is entered.

You should cancel the APPEND /X command before running the BACKUP or RESTORE commands, by typing:

```
APPEND ;
```

at the DOS prompt, and pressing the <ENTER> key.

NOTE: You can only set the /X option the first time you execute APPEND after powering on the system, or after restarting it by pressing <Ctrl-Alt-Del>.

/E

Stores the APPEND path listing in the DOS environment (like the PATH command).

When you store the path listing in the DOS environment, you can view or change the list by using the APPEND and SET commands. If you do not store the path listing in the DOS environment, you can only view or change the list by using the APPEND command.

NOTE: You can only set the /E option the first time you execute the APPEND command after powering on the system, or after restarting it by pressing <Ctrl-Alt-Del>.

APPEND

COMMENTS: The environment affected by the APPEND command is limited to the current command processor. If you load another command processor, or exit the current processor, the changes to the environment are lost. If you are using multiple processors, ensure that all processes are affected by changes to the APPEND path listing, by not using the /E option.

IMPORTANT

The path created by APPEND only applies to the reading of files. If the application program you are using also modifies and writes a file, the modified version of the file is most likely written to the default directory. This results in the unmodified original file existing in its original location on the disk drive, and the modified version of the file, with the same file name and type, existing on the default directory.

There is no way to remove the memory resident APPEND program from the PC, other than restarting the system.

ASSIGN

TYPE: External

PURPOSE: Instructs DOS to use an alternate disk drive for the one specified by a command or program.

FORMAT: [d:][pathname]ASSIGN [d1[=]d2[...]]

WHERE: [d:][pathname]

Indicates the drive and directory path where the ASSIGN command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1

Indicates the letter of the disk drive that the program or DOS normally uses.

d2

Specifies the letter of the disk drive you want to replace d1.

COMMENTS: When using this command:

- To remove the ASSIGN drive reassignment, type ASSIGN with no options.
- Do not enter a colon after the drive designation.
- You can set multiple reassignments at one time (i.e., ASSIGN A=C B=C assigns files from A and B to hard disk C).
- If you try to reassign a valid disk drive to an invalid drive, an INVALID PARAMETER error message appears. If this occurs, repeat the ASSIGN command with a valid drive.
- This command should be used primarily when you run applications that have been programmed to operate on specific drives. Your PC may not have the same set of drives, or you may want to run applications programmed for diskettes on your hard disk.

When you develop applications for possible sale or for use by others, you should minimize the need for the ASSIGN function by asking the user to specify drive assignments.

IMPORTANT

When a program asks the operating system to read or write a certain drive at a specific location, the ASSIGN command cannot change the access to another disk.

ATTRIB

TYPE: External

PURPOSE: Sets or removes the read attribute of a read-only file or the archive attribute of an archived file, or displays the current status.

FORMAT:
```
[d:][pathname]ATTRIB [+R | -R] [+A | -A]
[d1:][pathname1]filename1[.ext1]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the ATTRIB command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

+R

Sets the read attribute of the designated file to read only.

-R

Removes the read-only attribute.

+A

Sets the archive attribute of the designated file to archived.

-A

Removes the archived attribute.

[d1:][pathname1]

Indicates the drive and directory path of the read-only file.

filename1[.ext1]

Indicates the file name and extension of the read-only file (you can use the wildcard characters * and ? in this field).

ATTRIB

COMMENTS: A read-only file can only be read; it cannot be written to or erased. Setting the attribute of a file to read-only is useful if you want to ensure that the contents of the file are not accidentally erased or overwritten.

The archive attribute is used by other DOS commands, such as XCOPY.

EXAMPLES: In the following example, the attribute of the file C:\MYFILE.TXT is set to read-only:

```
ATTRIB +R c:\myfile.txt
```

To check for the attribute, type:

```
ATTRIB c:\myfile.txt
```

The following appears:

```
R C:\myfile.txt
```

where R indicates that the file is read-only.

AUTOEXEC.BAT

TYPE: Batch

PURPOSE: A file containing a series of commands to be automatically executed when you power on the PC.

FORMAT: AUTOEXEC.BAT

COMMENTS: AUTOEXEC.BAT is a batch file that can contain any of the seven batch commands shown below. The AUTOEXEC.BAT file can also contain any DOS command, or any command you have created. If the file exists at system startup, DOS executes each of the batch commands in sequence.

A typical AUTOEXEC.BAT file might contain a PROMPT command to change the DOS system prompt to the current directory name, and a PATH command to indicate where DOS can locate command files. The DATE and TIME commands are included as well, because there is no automatic prompt for the date and time if an AUTOEXEC.BAT file exists. For example:

```
PROMPT $P$G
PATH C:\MSDOS
DATE
TIME
```

Like any other batch file, the following commands can be included in the AUTOEXEC.BAT file:

```
ECHO
FOR..IN..DO
GOTO
IF
PAUSE
REM
SHIFT
```

For more information about the use of batch commands and AUTOEXEC.BAT, refer to the *MS-DOS User's Guide* (Order No. HU94).

BACKUP

TYPE: External

PURPOSE: To copy one or more files from a diskette or hard disk drive (or the DOS partition of the hard disk), to another diskette or hard disk drive.

FORMAT:
```
[d:][pathname]BACKUP
d1:[pathname1][filename1[.ext1]]
d2:[/S][/M][/A][/D:date][/T:time][/L:[filename]]
[/F]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the BACKUP command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1

Specifies the source disk drive to be backed up.

d2

Specifies the target disk drive for backup files.

[d1:][pathname1]

Indicates the drive and directory path of the source files.

filename1[.ext1]

Indicates the file name and extension of the source files (you can use the wildcard characters * and ? in this field).

/S

Copies all files in all of the subdirectories, in addition to the files in the designated directory.

/M

Copies all files that have changed since the last backup.

/A

Adds the files to a previous backup disk. If the target disk is not a previous backup, it is first purged of existing files and programs. When backing up to a diskette, you must use the last diskette from the previous backup; otherwise, your previous backup is written over.

/D:

Backs up all files created or modified since the given date (mm-dd-yy for USA format). Refer to the COUNTRY command description for the date format of other countries.

/T:

Backs up only files created on or after the time specified, as in the following example:

<time> = hh:mm A | P

where (USA time):

hh = hours 0-12
mm = minutes 0-59
A = AM
P = PM

Refer to the COUNTRY command description for the time format of other countries.

/L:

Creates a backup log with the file name specified on the source drive. If you do not specify a file name, DOS creates a file with the name BACKUP.LOG on the source drive.

/F

Formats the target disk if it is not already formatted.

COMMENTS: Some things to remember when using BACKUP:

- The source of a BACKUP can be the entire diskette or hard disk DOS partition, one or several specified files, a directory path, or any combination of the above.

- The source and target disks can be any diskette or hard disk drive.

- The default source drive is the current drive. Also, if you do not supply any parameters with the source drive, the default source to be backed up is the current directory only. Use the pathname to specify other directories for the backup, and the /S option if you want to back up the files within the subdirectories of the current directory.

- The target disk(s) must have been formatted under DOS prior to executing BACKUP, unless you specify the /F option.

- BACKUP deletes all files currently on the target disk or directory. (The only time the target is not erased is when you use the /A option to append the new files to the end of the last disk of your previous backup.)

  NOTE: The /A option only works in conjunction with the last backup diskette where multiple diskettes are required to perform the backup.

- When the target disk is a hard disk, the backup files are put in a \BACKUP subdirectory of the root directory.

- The BACKUP command prompts you to insert the next diskette in the target drive when the current diskette is full. BACKUP numbers these diskettes so that when you are RESTORING the files, it can verify that you have inserted the diskettes in the correct order.

- If you use the /L option, but do not specify a file name, a log file is created in the root directory of the source drive with the name BACKUP.LOG. The first entry in this log file is the date and time the file was created. Each subsequent entry contains the number of the target disk where the file was written (1, 2, 3, etc.) and the name of the backup file. If a backup log already exists in the root directory, the new backup log entries are appended to the existing backup log.

EXAMPLES:

```
BACKUP  C:  A:
```

Backs up the current directory of C to diskette drive A. If subdirectories exist in the current directory of C, they are ignored by the backup.

```
BACKUP  A:  B:/S/L
```

Backs up the current directory files of drive A, plus any of its subdirectories and the files in them. If the current directory is the root, then the entire diskette is backed up. A backup log file is created in the current directory of drive A, with the name BACKUP.LOG.

```
BACKUP  C:\  A:/A/M
```

Backs up all files in the root directory of drive C that have changed since the last backup was performed. These new backup files are appended to the end of the last backup if the last diskette from the previous backup is put in drive A. If any other diskette is put in drive A, the files are overwritten.

```
BACKUP  C:\LEVEL2\*.COM  A:/S/D:01-01-84
```

Backs up all files with the .COM extension and a file date of Jan 1, 1984 or later in the LEVEL2 directory, and all of its subdirectories.

```
BACKUP  C:\LEVEL2\*.COM  A:/S/T:05:00P
```

Backs up all files onto drive A with the .COM extension and a file creation time of 5:00 PM or later that are located in the LEVEL2 directory of drive C, and all of its subordinate directories.

DOS sets an exit code after completing the BACKUP command. For example, 0 indicates successful completion, 1 indicates that there are no eligible files to backup on the designated source drive, 2 indicates a file sharing conflict, 3 indicates that the backup was terminated by the user (using a <Ctrl-Brk>), and 4 indicates that the backup was terminated because of an unrecoverable error. Any other number also means that the backup was not successful.

If the backup is not successful, check the BACKUP command format, make sure the target drive is closed and a formatted, nonwrite-protected diskette is in the drive. Then retry the command. You can test the exit code by using the "IF command ERRORLEVEL number" condition in a batch file. Refer to the IF command description, described later in this section.

Use the RESTORE command to retrieve the files that have been backed up.

To check for accuracy while writing the disks, set VERIFY to ON before doing a BACKUP or RESTORE (refer to the VERIFY command description, later in this section).

NOTE: Do not use the BACKUP command while the ASSIGN, JOIN, or SUBST commands are in effect. The backup files may not be able to be restored.

BREAK

TYPE: Configuration or Internal

PURPOSE: To specify a more frequent test for a <Ctrl-Break> or <Ctrl-C> entry.

FORMAT: For configuration:

BREAK=[ON | OFF]

For internal:

BREAK [ON | OFF]

COMMENTS: With BREAK OFF, which is the default condition, DOS checks for entry of a <Ctrl-Break> sequence during the execution of a program only when accessing either a standard input, output, or printer device or a serial/parallel communications controller. A compiler or other program that performs few such accesses is difficult to interrupt.

When BREAK is ON, DOS tests for <Ctrl-Break> any time DOS is entered for any reason by the executing program.

If you are running an application program that uses Ctrl-Break function keys, turn off the function, so that when you press <Ctrl-Break> you affect your program and not the operating system. Because DOS is set with BREAK=OFF when the operating system is loaded (unless BREAK=ON is in CONFIG.SYS) there is no need to set the BREAK=OFF unless it has been previously set to BREAK=ON. BREAK is an interactive command that you can use in a batch or configuration file, or issue from the command line.

If you issue BREAK with no parameters, the current state of the BREAK command is displayed.

BUFFERS

TYPE: Configuration

PURPOSE: To specify the number of buffer storage areas that DOS uses.

FORMAT: BUFFERS=nn

WHERE: nn

Specifies any integer between 1 and 99 inclusive.

COMMENTS: A buffer is an area of system memory used to store a block (usually several records) of data read from, or to be written to, a disk. If you do not use the BUFFER command, DOS defaults to the following values:

System	Buffers
Base system	2
If any disk is > 360K	3
Main memory = 128 - 255 KB	5
Main memory = 256 - 511 KB	10
Main memory => 512 KB	15

Normally, these values are sufficient for most programs. However, if an application requires a lot of reading and writing to and from the disk, a greater number of buffers usually speeds up execution. DOS can then access the information directly, instead of searching and reading the disk.

There can be too few or too many buffers for a program. Too few buffers can cause DOS to spend too much time accessing the disk, since less data is stored in RAM memory. Too many buffers can slow DOS, due to the time spent handling buffers instead of executing a program. For larger programs, accounting applications and word processing, 12 to 24 buffers usually suffice.

CALL

TYPE: Batch

PURPOSE: Allows a batch file to execute another batch file. After the second batch file runs, control is returned to the original batch file.

FORMAT: CALL [d:][pathname]filename

WHERE: [d:][pathname]

Indicates the location of the second batch file.

filename

Specifies the name of the second batch file. The .BAT extension is not included in the specification.

COMMENTS: You can make recursive calls (a batch file can call itself).

CHCP

TYPE: External

PURPOSE: Selects the code page for DOS to use for as many devices as possible. CHCP is a system level command, while the MODE command (formats 5, 6, 7, and 8) is device level command.

FORMAT: To display the current code page:

[d:][pathname]CHCP

To change to another code page:

[d:][pathname]CHCP nnn

WHERE: [d:][pathname]

Indicates where the CHCP utility is located.

nnn

Specifies the number of the code page to which you would like to switch.

EXAMPLES: In order to allow the CHCP command to operate, the code pages must have been already prepared. In the following example, a PC that boots from hard disk drive C: (with DOS files in directory C:\DOS, and an EGA video display board) can be switched between code pages 437 and 850:

in C:\CONFIG.SYS:

```
DEVICE=c:\dos\ansi.sys
COUNTRY=001, 437 c:\dos\country.sys
DEVICE=c:\dos\display.sys con:=ega,437,2)
```

in C:\AUTOEXEC.BAT:

```
MODE CON CP PREP=((437,850) c:\dos\ega cpi)
MODE CON CP SEL=437
NLSFUNC c:\dos\country.sys
```

You can then display the code pages available for the CONsole (monitor) with the command:

```
MODE CON CP
```

Refer to Appendix D in the *MS-DOS User's Guide* (Order No. HU94) for instructions on using code page switching.

NOTE

You must load the NLSFUNC command before you issue the CHCP command.

CHDIR (CD)

TYPE: Internal

PURPOSE: Changes the current (working) directory to a different directory or displays the name of the current (working) directory.

FORMAT: CHDIR [d:][pathname]

or

CD [d:][pathname]

WHERE: [d:][pathname]

Specifies the drive and directory path to which you want to change.

EXAMPLES: If your working directory is \USER\CHUCK, for example, and you want to change to another directory, such as \USER\CHUCK\FORMS, type:

```
CHDIR \USER\CHUCK\FORMS
```

or

```
CHDIR FORMS
```

and press the <ENTER> key.

The following example, using the short version of the CHKDIR command, puts you in the parent directory of your working directory:

```
CD ..
```

The following example always places you in the root directory of the current disk:

```
CD \
```

When you use CHDIR without a pathname, it displays your working directory.

When you specify a drive other than the current one in a CHDIR command, the current directory on the second drive is displayed, not necessarily the root directory. For example:

```
A>CD C:
```

CHKDSK

TYPE: External

PURPOSE: Scans the default or specified disk drive, checks it for consistency, reports disk and memory capacities, and corrects disk errors.

FORMAT:
```
[d:][pathname]CHKDSK
[d1:][filename1[.ext1]][/F][/V]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the CHKDSK command is located. This option is not necessary if the command is in the current directory or you have previously defined a path to the command.

d1:

Specifies the drive to be checked.

filename1[.ext1]

Specifies the file name and extension of the file to be checked for noncontiguous blocks.

/F

Requests correction of disk errors.

/V

Displays pathnames and file names while running.

COMMENTS: Run CHKDSK occasionally on each disk to check for lost cluster errors in the directory. If errors are found, CHKDSK displays error messages, and a status report similar to the following:

```
21309440 bytes total disk space
  163840 bytes in 11 hidden files
    2048 bytes in 1 directories
  608256 bytes in 57 user files
   81920 bytes in bad sectors
20453376 bytes available on disk

  524288 bytes total memory
  461360 bytes free
```

CHKDSK displays the lost cluster errors found in the specified directory and gives you the option of correcting them. You cannot actually change anything, however, unless you specified the /F (fix) option when you typed the CHKDSK command. The /V option causes CHKDSK to display pathnames and file names while it is running.

If you enter a file name in addition to the disk drive designator, DOS checks the entire disk and produces the status report, and additionally, reports any noncontiguous blocks that exist for the specified file name. You can then use the COPY command to collect any noncontiguous blocks into one block.

CLS

TYPE: Internal

PURPOSE: Clears the screen.

FORMAT: CLS

COMMENTS: CLS (clear screen) removes all characters from the screen. The cursor is returned to the upper left-hand corner of the screen. The CLS command does not affect disk files or memory in any way.

If you have used the ANSI control codes to set the foreground and background colors, they remain the same.

If the foreground/background color control codes have not been set, the screen reverts to light foreground and dark background.

COMMAND

TYPE: Internal

PURPOSE: Loads a secondary version of the command processor.

FORMAT: COMMAND [d:][pathname][/P][/C string][/E:nnnnn]

WHERE: [d:][pathname]

Specifies the drive and directory path of the secondary command processor.

/P

Retains the second command processor permanently. With this option you cannot exit the secondary command processor until you restart the system.

/C

Causes the following DOS command line (string) to be executed by the secondary command processor, followed by an automatic exit back to the primary command processor.

string

Specifies the DOS command to be executed by the secondary command processor.

/E:nnnnn

Specifies the environment size, where nnnnn is the size in bytes. The size may range from 128 to 32768 bytes, with a default value of 160 bytes.

NOTE: If you set both the /P and /C switches, the /P switch is ignored.

COMMENT: COMMAND tells DOS to utilize a secondary version of the COMMAND.COM file from the disk and directory that you specify. For example, the secondary version of COMMAND.COM can be a copy of the standard COMMAND.COM in which you have changed the programming environment using the SET command, or added a PROMPT command to redefine the cursor. The secondary command processor environment changes remain in effect until you exit to the primary command processor. The COMMAND.COM file within DOS handles your housekeeping activities, your internal DOS commands, your system prompt, and your error handling routines. Do not attempt to load COMMAND.COM files from any other operating system or from another version of DOS with this command.

COMP

TYPE: External

PURPOSE: Compares two files, or sets of files, and reports their differences.

FORMAT:
```
[d:][pathname]COMP
[d1:][pathname1][filename1[.ext1]]
[d2:][pathname2][filename2[.ext2]]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the COMP command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

[d1:][pathname1]

Specifies the drive and directory path of the first (primary) file(s) you want to compare.

filename1[.ext1]

Specifies the file name and extension of the primary file(s).

[d2:][pathname2]

Specifies the drive and directory path of the second (secondary) file(s) you want to compare.

filename2[.ext2]

Specifies the file name and extension of the secondary file(s).

COMMENTS: The COMP command is useful to check a recently copied file. Some things to remember about the COMP command:

- You can use wildcard characters (? and *) in the command line.
- You cannot compare files of different lengths. If you attempt to do this, an error message appears.
- Upon exceeding ten mismatched characters when comparing two files, the COMP command aborts.
- If you specify a drive or pathname without a file name, the COMP command assumes a file name of *.*, and compares all of the files on the specified drive or in the specified directory to the corresponding secondary files.
- If you enter COMP without any options, the command prompts you for the file names to be compared. Use the command without options when you must change diskettes (such as comparing files on two diskettes with only one disk drive).

  If your files are identical, the following message appears:

  ```
  File compare OK
  Compare more files (Y/N)?
  ```

  If there is an inequality between the files, the following message is displayed:

  ```
  Compare error at offset nnnn
  File1 = xx
  File2 = yy
  ```

The offset number refers to the number of bytes from the beginning of the file to the location where the error occurred (using base 16). The next two lines show the hexadecimal code of the offending characters. For example, if at the beginning of a file (file1), the word "the" was transposed as "hte" in file2, the first error message would read:

```
Compare error at offset 01
File1 = 74
File2 = 68
```

and the second would read:

```
Compare error at offset 02
File1 = 68
File2 = 74
```

CONFIG.SYS

TYPE: Configuration

PURPOSE: A file containing commands to configure the DOS system. The commands are executed at system startup, before the DOS COMMAND.COM file is given control.

FORMAT: CONFIG.SYS

COMMENTS: The CONFIG.SYS file contains commands that configure or set up system parameters. These parameters must be established before the DOS system begins to interact with batch and interactive system commands. This is why the configuration commands are executed prior to giving control to the DOS COMMAND.COM program.

Only the following DOS configuration commands can be contained in the CONFIG.SYS file:

BREAK
BUFFERS
COUNTRY
DEVICE
FCBS
FILES
LASTDRIVE
SHELL
STACKS

For more information about the use of the configuration file, refer to Section 8.

COPY

TYPE: Internal

PURPOSE: Copies one or more files to the same or another disk. You can give the copies different names, and optionally combine several files into one larger file.

FORMAT: COPY [d1:][pathname1]filename1[.ext1][/A][/B]
[d2:][pathname2][filename2[.ext2]][/A][/B][/V]

WHERE: [d1:][pathname1]

Specifies the drive and directory path of the source file.

filename1[.ext1]

Specifies the file name and extension of the source file. The wildcard characters (* and ?) are permitted.

[d2:][pathname2]

Specifies the drive and directory path of the target file.

filename2[.ext2]

Specifies the file name and extension of the target file.

/A (source)

Treats file as an ASCII (text) file, and copies up to, but not including, the end-of-file marker (Ctrl-Z). The remainder is not copied. May also be used with pathname.

/A (target)

Adds an end-of-file character. May also be used with pathname.

/B (source)

Treats file as if it were binary, including end-of-file character. May also be used with pathname; copies entire file.

/B (target)

Does not add end-of-file character to binary file. Can also be used with pathname.

/V

Verification option.

COMMENTS: If you do not specify a target, the copy is to the default drive, and has the same name as the source file.

If the source is on the default drive and you do not specify the target, the COPY is aborted (copying files to themselves is not allowed).

When you copy a read-only file, the new file is not marked read-only. The default copy assumes the /B option, except as noted later in the discussion of concatenation.

The target diskette can have three forms:

- If you enter only a target drive designation (d2:), the source file is copied with the original file name to the designated drive.
- If you enter a target file name only, the source file is copied to a file on the default drive with the target file name specified.
- If you enter a full target file name, the source file is copied to a file on the designated drive with the target file name specified.

NOTE: If the target file already exists, it is overwritten, with the new copy of the file replacing the original.

The /V option verifies that the sectors written on the target disk are recorded properly. This option causes the COPY command to run more slowly because DOS must check each entry recorded on the disk. This option is not valid while networking is in effect, and is redundant if VERIFY=ON is in effect.

The COPY command does not copy hidden files. For this reason, refer to the SYS command description for instructions on copying the DOS hidden system files to a disk.

The COPY command also allows you to combine (concatenate) files when you copy. To concatenate files, list any number of files as options to COPY, separated by a plus sign (+), as in the following example:

```
COPY  FILE1.XYZ+FILE2.COM+B:FILE3.TXT BIGFILE.CRP
```

This command concatenates files named FILE1.XYZ, FILE2.COM, and FILE3.TXT (on drive B), and places them in sequence in the file BIGFILE.CRP (on the default drive). If BIGFILE.CRP were not specified above, the other three files would be concatenated into FILE1.XYZ by default. Concatenation assumes text files and the /A option. To concatenate binary files, use the /B option explicitly.

The following example combines several files into one file by using wildcard characters:

```
COPY  *.TXT  COMBIN.XMT
```

This command combines all files with a file name extension of .TXT into a file named COMBIN.XMT.

In the following example, each file matching *.WKD, is combined with the corresponding .REF file. The result is a file with the same file name, but with the extension .WK. Thus, MON.WKD is combined with MON.REF to form MON.WK, then TUE.WKD and TUE.REF form TUE.WK, and so on.

```
COPY  *.WKD+*.REF  *.WK
```

The following command combines all files matching *.FST, and all files matching *.SEC, into one file named COMBIN.TOT:

```
COPY  *.FST+*.SEC  COMBIN.TOT
```

Do not enter a concatenation COPY command where one of the source file names has the same extension as the target. For example, the following command is an error if ALL.LST already exists:

```
COPY  *.LST  ALL.LST
```

The error would not be detected, however, until ALL.LST is appended. (At this point it could have already been destroyed.)

The COPY command does, however, allow you to combine files using the following reversal of the above notation:

```
COPY  ALL.LST+*.LST
```

This command appends all *.LST files, except ALL.LST itself, to ALL.LST. This command does not produce an error message, and is the correct way to append files using the COPY command.

You can use the reserved device names (CON, LPT1, LPT2, AUX, etc.) in a COPY command. For example:

```
COPY  CON: AUTOEXEC.BAT
COPY  CON: CONFIG.SYS
```

These examples allow you to type information into the specified files. To use the COPY command to enter information into files:

1. Type the COPY command in the above format

2. Type the text

3. Press the <F6> key

4. Press the <ENTER> key.

The current contents of the file (if any) are erased.

The following examples display the contents of a file to the appropriate device:

```
COPY  FILE1  LPT1:
COPY  FILE1  CON:
```

The first example displays the contents of the file on the printer designated as LPT1. The second example, displays the contents of the file on the monitor.

Copying a file to another place with the same name changes the date and time saved with the new file:

The following command copies FILE1 from drive B to the default directory.

```
COPY  B:FILE1+
```

A command similar to the following changes the date for a file without moving it:

```
COPY  B:FILE1+,,  B:
```

In the following special case, if wildcard file name characters are used in the file name or file extension, then all of the matching files are appended together into the first file name that matches. The following command does not change the date and time of the files on drive B, but appends all of the files on drive B into a single file that replaces the first file found on drive B:

```
COPY  B:*.*+,,  B:
```

COUNTRY

TYPE: Configuration

PURPOSE: Selects the date, time, capitalization, collating sequence, folding format, and currency formats for a specific country.

FORMAT:

```
COUNTRY=ccc [ddd] [d:]filename[.ext]

or

COUNTRY=ccc [ddd]
```

WHERE: ccc

Indicates the three-digit international country code (the telephone area code) for the respective country's telephone system)

ddd

Specifies the code page of the desired country information. Each country has two code pages (Refer to Appendix D in the *MS-DOS User's Guide* (Order No. HU94) to select a specific code page).

[d:]filename[.ext]

Specifies the COUNTRY information file. The default name of this file is COUNTRY.SYS. You can call it by another name, however. You should enter a complete pathname for this file, as in the following example:

```
C:\SYSTEM\COUNTRY.SYS
```

COUNTRY

COMMENTS: The COUNTRY command allows you to change the date and time formats selected when making a backup copy of DOS Master Diskette 2 with the SELECT command. The command also changes the corresponding currency symbol and decimal separator. For example, if you choose the U.S. country code, the date format is mm-dd-yy and the time format hh:mm:ss; the period is used as decimal separator, and the currency symbol is a dollar sign ($) sign.

If you do not place a COUNTRY= statement in the CONFIG.SYS file:

- The default country code is 001.
- The default code page is 437.
- The default country information file is \COUNTRY.SYS.

Refer to the CHCP command description in this section for an example of how to integrate the COUNTRY command into your system.

The following table shows the country codes supported by DOS:

Country	Code	Date Format	Time Format
UNITED STATES	001	M-D-Y	12H hh:mm:ss
CANADIAN-FRENCH	002	Y-M-D	24H hh:mm:ss
LATIN AMERICA	003	D/M/Y	24H hh:mm:ss
NETHERLANDS	031	D-M-Y	24H hh:mm:ss
BELGIUM	032	D/M/Y	24H hh:mm:ss
FRANCE	033	D/M/Y	24H hh:mm:ss
SPAIN	034	D/M/Y	24H hh:mm:ss
ITALY	039	D/M/Y	24H hh:mm:ss
SWITZERLAND	041	D.M.Y	24H hh.mm.ss
ENGLAND	044	D-M-Y	24H hh:mm:ss
DENMARK	045	D-M-Y	24H hh.mm.ss
SWEDEN	046	Y-M-D	24H hh.mm.ss
NORWAY	047	D/M/Y	24H hh.mm.ss
GERMANY	049	D.M.Y	24H hh.mm.ss
AUSTRALIA	061	D-M-Y	24H hh:mm:ss
PORTUGAL	351	D/M/Y	24H hh:mm:ss
FINLAND	358	D.M.Y	24H hh.mm.ss
MIDDLE EAST	785	D/M/Y	12H hh:mm:ss
ISRAEL	972	D M Y	24H hh:mm:ss

CTTY

TYPE: Internal

PURPOSE: Changes the input device from which commands are issued (TTY represents the console).

FORMAT: CTTY AUX | COM1 | COM2 | COM3 | COM4 | CON

COMMENTS: The CTTY command changes the device through which you communicate with DOS. AUX, COM1, COM2, COM3, COM4, and CON are standard device drivers through which this communication is accomplished. This command is useful if you want to change the device on which you are working. The command:

```
CTTY AUX
```

moves all command I/O (input/output) from the current device (the console) to the AUX port, such as a teleprinter. The command:

```
CTTY CON
```

moves I/O back to the console.

IMPORTANT

Do not move I/O from the console to a printer without a keyboard. If this happens, you are not able to respond to DOS (since there is no input from an output-only device), and the system becomes inoperative until it is reset.

DATE

TYPE: Internal

PURPOSE: To display, enter, or change the DOS internal system date. This date is recorded in the directory for any files you create or alter.

You can change the date from your keyboard or from a batch file. (DOS does not prompt you to enter the date at power on if you use an AUTOEXEC.BAT file, unless you include a DATE command in that file.)

FORMAT: DATE [mm-dd-yy] U.S. date format

or

DATE [dd-mm-yy] International date format

or

DATE [yy-mm-dd] International date format

COMMENTS: If you type DATE, DOS responds with the message (for U.S. format):

```
Current date is DAY MM-DD-YYYY
Enter new date (mm-dd-yy):
```

If you do not want to change the date as shown, press <ENTER>.

You can also type a particular date after the DATE command on the command line, as in:

```
DATE 7-4-86
```

You must enter the new date using numerals only. Letters are not permitted. The allowed options are:

```
mm = 1-12
dd = 1-31
yy = 80-99 or 1980-2099
```

(if not included, 19 is assumed)

You can separate the date, month, and year entries by hyphens (-), periods (.), or slashes (/).

DATE

DATE displays the date in the format corresponding to the country code you specified with the COUNTRY or SELECT commands for international versions of DOS.

Leaving a PC on for longer than 24 hours without activity can cause the date to be updated incorrectly.

NOTE: The DATE command affects the setting stored by battery-operated clocks in AP and SP Series PCs. You can also use the SETUPPC utility to set the battery-operated clock. (Refer to Appendix D for more information about SETUPPC.)

To permanently set other battery-powered clocks, use the software provided with the clock option.

DEBUG

TYPE: Utility

Refer to Section 7.

DEL

TYPE: Internal

PURPOSE: Deletes specified files.

FORMAT:
```
DEL [d:][pathname]filename[.ext]

or

DEL [d:]pathname
```

WHERE: [d:][pathname]

Specifies the drive and directory path of the file(s) to be deleted.

filename[.ext]

Specifies the file name and extension of the file(s) to be deleted.

COMMENTS: You can use DEL to delete one or more files. If the file name is *.* (meaning erase all files), the prompt "Are you sure? Y/N" appears. If you type a Y or y and press the <ENTER> key, all files are deleted. You can also type ERASE instead of the DEL command. To delete the directory pathname, refer to the RMDIR command description.

NOTE: If you only specify a pathname, the command assumes that all of the files in that directory path are to be deleted (as with *.*). You cannot delete hidden files without reformatting the disk. You cannot delete files that are marked as read-only.

DEVICE

TYPE: Configuration

PURPOSE: Defines a device driver file, and loads the file each time DOS is started.

FORMAT: DEVICE=[d:][pathname]filename[.ext]

COMMENTS: A device driver is the software used to control a peripheral. DOS includes the drivers needed for all standard hardware. When you start the system, the existing driver software is loaded so you can control the devices. However, if you add devices to your system, you need to identify the appropriate software through your CONFIG.SYS file, so that DOS can talk with the new hardware devices. This is done using the DEVICE command.

ANSI.SYS

One of the more common device driver programs is the ANSI.SYS file provided with DOS. The file allows you to expand the usage of a color display (by controlling the color through programs), or to reprogram the keyboard. ANSI.SYS replaces standard display and keyboard driver functions with a set of extended screen and keyboard functions.

To place the definition of the ANSI.SYS file within CONFIG.SYS, use EDLIN, another text editor, or the COPY CON: command to enter the following command in the CONFIG.SYS file:

```
DEVICE=[d:][pathname]ANSI.SYS
```

Where:

[d:][pathname]

Indicates the drive and directory where the ANSI.SYS file is located. This option is not necessary if the ANSI.SYS file is in the root directory.

DEVICE

If you are using a hard disk, and the ANSI.SYS file is located in a subdirectory, you must specify the directory path as well. In the following example, \SYS indicates that the ANSI.SYS file is located in the SYS directory (a subdirectory of the root directory).

```
DEVICE=C:\SYS\ANSI.SYS
```

Typically, diskette users place the device drivers in the root directory.

DISPLAY.SYS

DISPLAY.SYS allows you to implement code page switching on a Bull PC using the following displays:

- EGA
- VGA
- LCD (on the laptop PC)

It is necessary to use code page switching if you need to support multiple language character sets with your PC. Refer to Appendix D in the *MS-DOS User's Guide* (Order No. HU94) for information on code page switching.

NOTE: If you are also loading the ANSI.SYS device driver, it must precede the DISPLAY.SYS command in the CONFIG.SYS file.

To enable display code page switching, place the following command in the CONFIG.SYS file:

```
DEVICE=[d:][pathname]DISPLAY.SYS
CON[:]=(type[,[hwcp][,n]])
```

or

```
DEVICE=[d:][pathname]DISPLAY.SYS
CON[:]=(type[,[hwcp][,(n,m)]])
```

Where:

[d:][pathname]

Indicates the drive and directory where the DISPLAY.SYS file is located. This option is not necessary if the DISPLAY.SYS file is in the root directory.

type

Specifies the display adapter type (MONO, CGA, EGA, and LCD). PCs with VGA adapters are properly programmed as the EGA type.

hwcp

Specifies the code page supported by the display adapter. The possible values include 437, 850, 860, 863, 865, etc.

n

Specifies the number of additional code pages that can be supported by the display adapter. The allowable ranges, based on the display type, are listed in the following tables.

m

Specifies the number of sub fonts supported for each code page. If you do not specify m, the default value is used. Default values are shown in the following tables.

The following n and m parameters are available for the display adapter types listed:

Type	Adapter Type	Default n	Range Of n	Default m	Range Of m
CGA	Color Graphics	0	0	0	0
EGA	Enhanced Graphics	1	1-12	2	1-2
	VLSI Graphics[a]	1	1-12	2	1-2
LCD	Laptop Liquid Crystal Display	1	1-12	2	1-2
MONO	Monochrome/ Printer	0	0	0	0

[a]VGA adapters respond to the EGA command.

The following sub fonts are supported by the adapter "types" listed:

Type	Adapter Type	Default m	Font Size
EGA	Enhanced Graphics	2	8 X 8 8 X 14
	VLSI Graphics[a]	2	8 X 8 8 X 16
LCD	Laptop Liquid Crystal Display	1	8 X 8

[a]VGA adapters respond to the EGA command.

For example, to install code switching for the CON: device, include the following line in CONFIG.SYS:

```
DEVICE=C:\DOS\DISPLAY.SYS
CON:=(EGA,437,2)
```

This format also sets DISPLAY.SYS to support a machine with an EGA that has code page 437 built into it, and to expect to have two more code pages prepared for it. To prepare these pages, use the MODE command, format 5 (MODE ... CP PREP). Refer to the MODE and CHCP command descriptions in this section.

DRIVER.SYS

The DRIVER.SYS file allows you to assign a logical drive letter to a physical disk drive. To install DRIVER.SYS, include the following command line in your CONFIG.SYS file:

```
DEVICE=[d:][pathname]DRIVER.SYS /D:ddd
[/C][/F:f][/H:hh] [/N][/S:ss][/T:ttt]
```

Where:

[d:][pathname]

Indicates the drive and directory where the DRIVER.SYS file is located. This option is not necessary if the DRIVER.SYS file is in the root directory.

/D:ddd

Specifies a physical drive number. ddd can range from 0 to 255. Physical drives are numbered differently than logical drives. Diskette drives are numbered starting at 0, and hard disk drives are numbered starting at 128.

For example, if your PC has two diskette drives, they might be numbered 00 and 01. If you add an external diskette drive, its physical drive number is 02.

If your PC has one diskette drive and one hard disk drive, the diskette drive is number 00, and the hard disk drive is number 128. If you add an external diskette drive, its physical drive number is still 02, because DOS already knows the definitions of physical drives 00 and 01. If you add a second hard disk, its physical drive number is 129.

/C

Indicates that changeline (doorlock) support is required.

/F:f

Specifies the form factor index where f can be the following:

0 = 320/360 KB diskette drive
1 = 1.2 MB diskette drive
2 = 720 KB diskette drive
3 = 8-inch single density diskette drive
4 = 8-inch double density diskette drive
5 = hard disk drive
6 = tape drive
7 = other

If you do not specify the /F option, DRIVER.SYS uses a default of 720 KB.

/H:hh

Specifies the maximum head number. hh can range from 1 to 99.

/N

Indicates a nonremovable block device. A hard disk is an example of a nonremovable block device.

/S:ss

Specifies the number of sectors per track. ss can range from 1 to 99.

/T:ttt

Specifies the number of tracks per side on the block device. ss can range from 1 to 999.

For example, if you add an external 720 KB diskette drive to your PC, include the following line in the CONFIG.SYS file:

```
DEVICE=DRIVER.SYS /D:02
```

EMMDRV.SYS

The EMMDRV.SYS file supports applications that adhere to the Lotus/Intel/Microsoft (LIM) Expanded Memory Specification (EMS).

To install the EMMDRV.SYS, add the following command to your CONFIG.SYS file:

```
DEVICE=[d:][pathname]EMMDRV.SYS S Knnnn
```

Where:

[d][pathname]

Indicates the drive and directory where the EMMDRV.SYS file is located. This option is not necessary if the EMMDRV.SYS file is in the root directory.

S

Specifies the Software Emulation feature. Software emulation uses extended memory to emulate expanded memory.

Knnnn

Specifies the total number of kilobytes (KB) to be managed by the Extended Memory Manager (EMM). The value must be between 16 KB and 8192 KB (8 MB), and be a multiple of 16. To use extended memory installed on the system, Knnnn should equal extended memory, less 1024, rounded to the nearest 16K multiple. If the value does not meet these requirements, an error message is displayed, and the EMM is not installed.

PRINTER.SYS

The PRINTER.SYS file allows you to use code switching for parallel ports LPT1, LPT2, and LPT3 (you can use PRN in place of LPT1).

NOTE: The printer attached to your parallel port must support code page switching (the IBM Proprinter Model 4201 or compatible).

To install PRINTER.SYS, insert a command line in the following format in your CONFIG.SYS file:

```
DEVICE=[d:][pathname] PRINTER.SYS
LPT#=[type,[hwcp[,ccc]][,n]]
```

Where:

[d:][pathname]

Indicates the drive and directory path where the PRINTER.SYS file is located. This option is not necessary if the PRINTER.SYS file is in the root directory.

type

Indicates the printer in use:

- 4201 (for IBM Proprinter Model 4201 or compatible)
- 5202 (for IBM Quietwriter III Printer Model 5202 or compatible)

hwcp

Indicates the supported code page:

437 (United States)
850 (Multilingual)
860 (Portugal)
863 (French-Canadian)
865 (Norway)

n

Indicates the number of additional code pages that can be supported (maximum of 12).

Refer to the MODE command description for instructions on preparing and selecting code pages.

NOTE: Do not perform a MODE prepare or select while data is being printed by the PRINT command.

DEVICE

RAMDRIVE.SYS

The RAMDRIVE.SYS file allows you to maintain a drive in Random Access Memory (RAM). This file differs from VDISK in that a RAMDRIVE can be configured to run out of either extended or expanded memory. A VDISK can only run out of extended memory.

To create one or more RAM drives, include one or more of the following command format options in your CONFIG.SYS file (in the root directory of your boot disk).

```
DEVICE=[d:][pathname]RAMDRIVE.SYS [bbb][sss] [ddd]
[/A /E]
```

Where:

[d:][pathname]

Indicates the drive and directory path of the RAMDISK.SYS file.

[bbb]

Specifies the diskette size in kilobytes. The default is 64 KB.

[sss]

Specifies the sector size in bytes. Appropriate values are 128, 256, 512, and 1024 bytes. The default is 128 bytes.

[ddd]

Specifies the number of directory entries (from 4 to 1024). The default value is 64.

/A

Uses expanded memory to create the RAM disk.

/E

Uses extended memory to create the RAM disk.

NOTE: Because the execution of RAMDRIVE.SYS assigns the next available drive letter(s) to the RAM drives, it may be necessary to adjust the value of the LASTDRIVE= command, described in this section. The LASTDRIVE= command must precede the DEVICE=RAMDRIVE.SYS command in the CONFIG.SYS file in order to be effective.

SMARTDRV.SYS

The SMARTDRV.SYS driver allows you to maintain a copy in RAM of the most recent files that you have copied to or from the hard disk drive(s). This buffer is known as a disk cache.

SMARTDRV.SYS is of benefit primarily when running some of the MicroSoft Windows software packages. These packages use the boot disk as an overlay memory device. By storing the most recently used (program) data in memory, windows applications programs can run faster (the disk access time is greatly reduced).

NOTE: Installing a RAM drive or VDISK does not replace the value of SMARTDRiVe, as they do not replace the boot drive with a disk image in memory.

The larger the SMARTDRiVe becomes, the more time the system spends scanning it before accessing a specified file for the first time. Thus, you can slow down your PC with too large a disk cache.

To set up a disk cache, include a form of the following command in the CONFIG.SYS file, in the root directory of your boot disk:

```
DEVICE=[d:][pathname]SMARTDRV.SYS bbb [/A]
```

Where:

[d:][pathname]

Indicates the drive and path where the SMARTDRV.SYS file is located.

bbb

Specifies the buffer's size in kilobytes.

/A

Specifies use of expanded memory.

NOTE: Because a SMARTDRiVe does not assume any of the available disk drive letter(s), you do not need to adjust the LASTDRIVE= command, as you would when using a RAM drive or VDISK.

Refer to Appendix E in the *MS-DOS User's Guide* (Order No. HU94) for details on using SMARTDRV.SYS.

VDISK.SYS

The VDISK.SYS file allows you to maintain a virtual disk (VDISK). A VDISK is an area of memory that you can store files in and read files from, as you would on a physical disk drive.

Because the VDISK exists in memory, and does not spend time reading and writing to a physical drive, the VDISK is much faster than a physical drive. However, because the data stored in a VDISK exists in memory, all data on the VDISK is lost when you restart the system (<Ctl-Alt-Del>) or turn off the power. You must, therefore, save the VDISK files you wish to keep onto a physical diskette or hard disk before restarting or powering off the system.

To set up one or more VDISKs, first make sure that the file VDISK.SYS is present on the disk that you use to load DOS. If necessary, copy VDISK.SYS from your original DOS Master diskette.

To create a VDISK, include a form of the following command in the CONFIG.SYS file, in the root directory of your boot disk:

```
DEVICE=[d:][pathname]VDISK.SYS
[comment][bbb][comment][sss][comment]
[ddd][/E[:t]]
```

Where:

comment

Contains ASCII characters in the range of 20 through 126, except the slash /. For example:

```
DEVICE=c:\dos\VDISK.SYS buffer size=360
sector size=512 directory entries=112
```

bbb

Represents the size of the VDISK in kilobytes. The minimum VDISK size created by DOS is 1 KB. The default is 64 KB.

sss

Specifies the sector size in bytes. Permitted values are 128, 256, and 512 (default).

ddd

Specifies the number of directory entries that are to be allowed on the VDISK. The default value is 64. The range of values is between 2 and 512.

/E

Specifies the VDISK in extended memory for 80286 systems (i.e., above 1 MB). The <t> parameter represents the transfer limit. <t> is the number of sectors that VDISK copies to extended memory at one time. The transfer limit is a number between 1 and 8, with a default of 8. Because VDISK disables interrupts when moving sectors to extended memory, large values for sector size and transfer limit may cause loss of high speed I/O interrupts.

When you have finished editing CONFIG.SYS, save the new version. The next time you start the system, VDISK installs the virtual disk(s) in the order in which you specified them, and the following message is displayed:

VDISK Version 3.30 Virtual Disk x:

where x is the drive designation assigned to the virtual disk.

The virtual disks are assigned drive designations following those of the real disk drives. If your system has two diskette drives, the first (or only) virtual disk drive is referred to as drive C. If your system already has a drive C, the first virtual disk is regarded as drive D, the next as E, and so on.

If a VDISK cannot be installed, an error message is displayed, and the installation procedure is terminated.

After a successful virtual disk installation, VDISK informs you about the size of the VDISK, the space available for data (as seen via CHKDSK), the sector size, and the maximum number of sectors that can be transferred in one logical access.

```
Buffer size:  nnnn bytes
Sector size:  nnn bytes
Directory entries:  nnn bytes
Transfer Limit:  nn bytes
```

The first directory entry of a virtual disk contains a volume label which is useful in disk identification.

DIR

TYPE: Internal

PURPOSE: Displays all nonhidden directory entries in a specified directory, or only those of specified files.

FORMAT: DIR [d:][pathname][filename[.ext]][/P][/W]

WHERE: [d:]

Indicates the drive containing the entries you want to list.

[pathname]

Indicates the directory containing the entries you want to list.

[filename[.ext]]

Specifies the file(s) you want to list. You can use the wildcard characters (* and ?) to list groups of files.

/P

Causes a pause when screen is full. Pressing any key continues scrolling.

/W

Causes a wide (80 column) display of file names and directory names only. Does not include file size and other information.

EXAMPLES: If you just type:

```
DIR
```

and press the <ENTER> key, all nonhidden entries in the default directory are displayed.

If you add only the drive specification:

```
DIR d:
```

All entries in the current directory on the disk in the specified drive are displayed.

If you type only a file name, with no extension:

```
DIR filename
```

all files with the designated file name on the disk in the default directory are displayed.

You can use the wildcard characters (* and ?) in the file name option. Following are equivalent command designations.

```
DIR          =  DIR *.*
DIR FILENAME =  DIR FILENAME.*
DIR .EXT     =  DIR *.EXT
```

You can also redirect the directory display to a device other than the monitor. For example, to print the directory listing, type:

```
DIR > PRN
```

To place a copy of the directory listing into a file, type:

```
DIR > [d:][pathname]filename.ext
```

DISKCOMP

TYPE: External

PURPOSE: Verifies that two identical size and capacity diskettes contain identical data.

NOTE: This command does not compare hard disk drives or virtual disks. If you enter the drive letter of a hard disk drive or a virtual disk, an error message is displayed.

FORMAT: [d:][pathname]DISKCOMP [d1: [d2:]][/1][/8]

WHERE: [d:][pathname]

Indicates the drive and directory path where the DISKCOMP command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1

Specifies the first drive containing a diskette to be compared.

d2

Specifies the second drive containing a diskette to be compared.

/1

Compares only the first side of a diskette, even if it is a double-sided diskette.

/8

Compares only the first 8 sectors/track on a diskette containing 9/15 sectors/track.

COMMENTS: You normally execute the DISKCOMP command after you have created a duplicate diskette with the DISKCOPY command. DISKCOMP runs a comparison that is sector to sector and track to track. If the tracks are not equal, a message is issued that indicates the track number and side of the unequal track.

Since the COPY command only copies files, instead of duplicating sectors or tracks (that may be bad or empty), it is unlikely that a "copy" of a diskette is physically identical to the original.

You can type the DISKCOMP command without the disk drive designations. DOS then prompts you to insert and remove diskettes as necessary. If you do not specify d1 and d2, or you have a single disk drive, DISKCOMP does a single drive comparison. If you specify only d1, then d1 is assumed to be the default drive.

NOTE: Do not use the DISKCOMP command on disks defined with the ASSIGN, JOIN, or SUBST commands.

You cannot use the DISKCOMP command to compare a 3-1/2 inch diskette and a 5-1/4 inch diskette. The physical size of both the source and target must be exactly the same.

DISKCOPY

TYPE: External

PURPOSE: Makes an identical copy of a 5-1/4 inch or 3-1/2 inch diskette onto a diskette of the same size (including any hidden files). Copies only diskettes.

FORMAT: [d:][pathname]DISKCOPY [d1: [d2:]][/1]

WHERE: [d:][pathname]

Indicates the drive and directory path where the DISKCOPY command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1

Specifies the source diskette drive.

d2

Specifies the target diskette drive.

/1

Copies only the first side of a diskette.

COMMENTS: The first drive you specify is the source drive; the second drive is the target drive. The target disk is reformatted if it is not already formatted the same as the source diskette.

You can specify the same drive for both the source and target drives. If the designated drives are the same, or if no drives are designated, a single-drive copy operation is performed. You are prompted to insert the correct diskettes at the appropriate times. DISKCOPY waits for you to press any key before continuing.

After copying, DISKCOPY prompts:

Copy another diskette (Y/N)?

If you press Y, the next copy is performed on the drives that you originally specified (after you have been prompted to insert the proper disks). To end the copying, press N.

Before using DISKCOPY, consider the following command characteristics:

- If you omit both drive designations, or specify only the default drive as the source drive, a single-drive copy operation is performed on the default drive.

- If you omit the target drive, the default drive is used as the target drive.

- Both diskettes must have the same number of physical sectors and those sectors must be the same size. The target diskette is automatically reformatted to match the source diskette.

- If the diskettes to be copied are not identical in size, the copy is not completed and an error message appears.

- If you have any doubt about the success of the DISKCOPY, you can use the DISKCOMP command to verify the accuracy of the copy.

- Diskettes that have had extensive file creation and deletion activity can become fragmented, because disk space is no longer allocated sequentially. (The first free sector found is the next sector allocated, regardless of its location on the disk.)

- A fragmented disk can cause poor performance due to delays involved in finding, reading, or writing a file. Since DISKCOPY is a disk, track-for-track, sector-for-sector command, you may want to use the COPY *.* command, or XCOPY, when copying older diskettes. These commands copy your diskette file for file, and eliminate the fragmentation.

EXAMPLE: To copy the contents of a diskette in drive A to the diskette in drive B, type:

```
DISKCOPY A: B:
```

DISKCOPY automatically determines the number of sides to copy, based on the source diskette in the drive, unless you use the /1 option.

If disk errors occur during a DISKCOPY, DOS displays an error message. Refer to Appendix G for information on error messages.

NOTE: Do not use the DISKCOPY command on disks defined with the ASSIGN, JOIN, or SUBST commands.

DISPLAY.SYS

TYPE: Configuration

Refer to the DEVICE command description.

DOSINS

TYPE: Installation

Refer to Section 3 in the *MS-DOS User's Guide* (Order No. HU94).

DRIVER.SYS

TYPE: Configuration

Refer to the DEVICE command description.

DRVINS

TYPE: Installation

Refer to Appendix B in the *MS-DOS User's Guide* (Order No. HU94).

ECHO

TYPE: Batch

PURPOSE: Turns the batch echo feature on and off.

FORMAT: ECHO [ON | OFF | message]

WHERE: message

Represents the characters to be displayed when ECHO is ON.

COMMENTS: Normally, commands in a batch file are displayed (echoed) on the screen when they are seen by the command processor. ECHO OFF turns off this feature. ECHO ON turns the screen echo back on. The ECHO <message> is displayed even if ECHO is currently off.

If you do not specify ON, OFF, or <message>, the current setting of ECHO is displayed.

EDLIN

TYPE: Utility

Refer to Section 5.

EMMDRV.SYS

TYPE: Configuration

Refer to the DEVICE command description.

ENHKEY

TYPE: External

PURPOSE: ENHKEY supports an IBM RT or compatible keyboard.

FORMAT: ENHKEY [/F]

WHERE: [d:][pathname]

Indicates the drive and directory path where the command is located. This option is not necessary if the ENHKEY command is in the current directory, or you have previously defined a path to the command.

/F

Specifies the use of international character sets.

NOTE: If you install KEYB for non-U.S. keyboard support, you must install ENHKEY with the /F option. Otherwise, the foreign language character set is overwritten by the default U.S. character set used by ENHKEY.

ERASE

TYPE: Internal

PURPOSE: Deletes specified files.

FORMAT: ERASE [d:][pathname]filename[.ext]

or

ERASE [d:]pathname

WHERE: [d:][pathname]

Specifies the drive and directory path of the file(s) to be erased.

filename[.ext]

Specifies the file name and extension of the file(s) to be erased.

COMMENTS: You can use ERASE to delete one or more files. If the file name is *.* (meaning erase all files), the prompt "Are you sure? Y/N" appears. If you enter a Y, all files are erased as requested.

If you enter only the pathname, you erase all files within the specified directory. To erase a directory, refer to the RMDIR command description.

You cannot erase hidden files without reformatting the disk. You cannot erase files that are marked as read-only.

EXE2BIN

TYPE: External

PURPOSE: Changes an .EXE file into a memory image .COM or .BIN file.

FORMAT:
```
[d:][pathname]EXE2BIN
[d1:][pathname1]filename1[.ext1]
[d2:][pathname2][filename2[.ext2]]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the EXE2BIN command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

[d1:][pathname1]

Specifies the drive and directory path of the .EXE file.

filename1[.ext1]

Indicates the file name and extension of the .EXE file.

[d2:][pathname2]

Specifies the drive and directory path of the .COM file.

filename2[.ext2]

Specifies the file name and extension of the .COM file.

COMMENTS: The EXE2BIN command converts executable files to a binary form, allowing faster loading and execution by compressing the coded file. If you do not specify a disk drive or pathname, the default drive/pathname is assumed. You should ensure that the .EXE format file is properly formatted within the guidelines of DOS.

NOTE: The resident, or actual, code should not exceed 64 KB.

EXIT

TYPE: Internal

PURPOSE: Exits a secondary COMMAND.COM (the command processor) and returns to a previous level, unless the secondary command processor was made permanent by the use of the /P option of the COMMAND command.

FORMAT: EXIT

EXAMPLE: To look at a directory on drive B while running an application program (such as GW-BASIC), enter the GW-BASIC SHELL command to load the secondary COMMAND.COM. You now are able to issue DOS commands. You can type the DIR command, for example, and DOS displays the directory for the default drive. When you enter EXIT, you return to the previous level GW-BASIC application program.

FASTOPEN

TYPE: External

PURPOSE: FASTOPEN is a memory resident program that holds a copy of a drive's directory structure, including pointers to where the files are physically located on a drive. This is known as caching the directory.

FASTOPEN greatly reduces the amount of time it takes to locate files on a drive.

FORMAT: [d:][pathname]FASTOPEN d:[=nnn] ...

WHERE: [d:][pathname]

Indicates the drive and directory that contain the FASTOPEN file.

d:

Specifies the drive to be cached. You can make multiple entries.

nnn

Specifies the number of directory or file entries (subdirectory pointers and file descriptions) to hold for the specified drive (min=10, max=999, default=34).

COMMENTS: you can only load FASTOPEN once after you power on the system or restart the system using <Ctrl-Alt-Del>. If you cache multiple drives on the disk, they should all be specified the first time you run FASTOPEN. For example, to cache the directories of a system with hard disk drives C and D, type the following command:

```
FASTOPEN C:=100 D:=100
```

NOTE: The system does not allow you to cache the directory of a diskette, such as drive A or B.

FCBS

TYPE: Configuration

PURPOSE: Specifies the number of File Control Blocks (FCBS) that can be concurrently open when using file sharing.

FORMAT: FCBS=xxx,yyy

WHERE: xxx

Represents the maximum number of files opened by FCBS that can be open at any one time (in the range 1 to 255). The default value is 4.

yyy

Specifies the number of files opened by FCBS that DOS cannot close automatically if an application tries to open more than xxx files by FCBS. The first yyy files opened by FCBS are protected from being closed. The value yyy can be in the range 0 to 255, and should not exceed xxx. The default value is 0.

The value of xxx should be equal to, or greater than, the value of yyy. If xxx and yyy are equal, DOS cannot automatically close FCB files if a program tries to open more than xxx files. If an error message is displayed because of DOS closing FCB files, increasing the value of xxx may solve the problem.

COMMENTS: This command is of importance only if file sharing is loaded. If file sharing is not loaded, the number of files that can be concurrently open is not limited.

FDISK

TYPE: External

PURPOSE: Creates DOS partitions on a hard disk (fixed disk), and performs a variety of hard disk tasks.

FORMAT: [d:][pathname]FDISK

WHERE: [d:][pathname]

Indicates the drive and directory path where the FDISK command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

COMMENTS: FDISK allows you to:

- Create a primary or extended DOS partition.
- Create logical drives.
- Change the active partition.
- Delete a primary or extended DOS partition.
- Display information about a partition.
- Select the next hard disk to be partitioned (if your system contains more than one hard disk).

When you type FDISK and press the <ENTER> key, the FDISK option selection menu appears:

```
Fixed Disk Setup Program Version 3.30
(C) Microsoft Corp. 1987

FDISK Options

Current Fixed Disk Drive: 1

Choose one of the following:

    1.  Create DOS Partition
    2.  Change Active Partition
    3.  Delete DOS Partition
    4.  Display Partition Information
    5.  Select Next Fixed Disk Drive

Enter choice: [1]


Press ESC to return to DOS
```

Option 5 appears on the menu only if your PC contains more than one hard disk. When the menu appears, option 1 is automatically shown as the default selection in the "Enter choice:" field.

The screens that appear for each option also contain default selections. To select a default selection (in this case, option 1), press the <ENTER> key. To select one of the other options, enter the option number, and press the <ENTER> key. If you make a mistake, press the Esc key to return to the option selection menu.

After you make a selection, additional screens are displayed for the option you have selected. Make the desired selection from the options listed on each screen, and press <ENTER>.

NOTE: If your hard disk does not contain an active partition, a warning prompt appears, indicating that the disk is not bootable.

Refer to Appendix C in the *MS-DOS User's Guide* (Order No. HU94) for instructions on partitioning and formatting your hard disk.

FILES

TYPE: Configuration

PURPOSE: Designates the number of files (handles) that can be open concurrently.

FORMAT: FILES=nnn

WHERE: nnn

Indicates the number of files (between 8 and 255, inclusive)

COMMENTS: The default number of files that can be open at any time is 8. If you specify a number below 8, DOS still allows you to open 8.

FIND

TYPE: External

PURPOSE: Searches for a specific string of text in one or more files, and displays each occurrence of the string.

FORMAT: [d:][pathname]FIND [/V][/C][/N]"string"
[[d1:][pathname1]filename1[.ext1]...]

WHERE: [d:][pathname]

Indicates the drive and directory path where the FIND command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

[d1:][pathname1]

Specifies the drive and directory path of the file to be searched.

filename1[.ext1]

Specifies the file name and extension of the file to be searched.

/V

Displays all lines that do not contain the string.

/C

Counts the number of occurrences of the string in the file without displaying each occurrence.

/N

Displays the line number in front of the string.

"string"

Specifies the group of characters (within double quotes) that you want to find.

NOTE: You cannot use the wildcard characters * and ? with the FIND command.

COMMENTS: FIND is a filter that searches the specified file(s) for the specified string.

The FIND command optionally displays:

- All file lines that contain the specified string
- All file lines that do not contain the specified string
- A count of the number of occurrences of the specified string
- The line numbers of all file lines containing the specified string.

If you do not specify any files, FIND takes the input on the screen and displays all lines that contain the specified string.

You must enclose the string in quotes. For example, if you type:

```
FIND "Fool's Paradise" BOOK1.TXT BOOK2.TXT
```

DOS displays all lines from BOOK1.TXT to BOOK2.TXT (in that order) that contain the string "Fool's Paradise."

If you type:

```
DIR B: | FIND /V "DAT"
```

DOS displays all of the names of the files on the disk in drive B that do not contain the string "DAT". Type double quotes around a string that already has quotes in it.

FOR..IN..DO

TYPE: Batch

PURPOSE: Allows repetitive execution of DOS commands.

FORMAT: For batch processing:

FOR %%variable IN (list) DO command [%%variable]

For interactive processing:

FOR %variable IN (list) DO command [%variable]

WHERE: variable

Indicates any noncapitalized single letter.

(list)

Specifies the list of characters to substitute for <variable>. You must enclose the entire list in parentheses.

command

Specifies any DOS system command or batch subcommand.

NOTE: You must enter the space between IN and (list). This space was optional in earlier versions of DOS.

COMMENTS: This command allows you to:

- Select certain items (IN<space>(list))
- Define this selection as one variable (FOR variable)
- Perform a DOS command (DO command) on each of your selections (variable) in sequence.

The <variable> parameter can be any character except 0, 1, 2, 3, ..., 9 (to avoid confusion with the %0 to %9 batch processing replacement parameters). Use the %% in front of the <variable> for the FOR..IN..DO command that is included in a batch processing file, and use the % in front of the <variable> for issuing the command from the DOS command line (interactive processing).

The following command, for example, can be used interactively:

```
FOR %H IN (HELP DOS) DO DIR %H*.*
```

This command lists all files or directories beginning with the letters HELP, and then beginning with the letters DOS.

The following example command can be entered in a batch file:

```
FOR %%F IN (FILE1.EXT FILE2.EXT FILE3.EXT)
DO COPY %%F C:
```

This command copies the files FILE1.EXT, FILE2.EXT, and FILE3.EXT from the default diskette drive to the C drive each time the batch file is run.

FORMAT

TYPE: External

PURPOSE: Formats the disk in the specified drive to accept DOS files.

FORMAT:
[d:][pathname]FORMAT
[d1:][/S][/V][/1][/4][/8][/B][/N:xx][/T:yy]

WHERE: [d:][pathname]

Indicates the drive and directory path where the FORMAT command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1

Specifies the drive to be formatted.

/S

Copies DOS system files to the newly formatted disk.

/V

Allows you to enter a volume label (name).

/1

Formats one side of the diskette only (5-1/4 inch drives only).

/4

Formats nonhigh-density diskettes in a high capacity drive (5-1/4 inch drives only).

/8

Formats diskette with eight sectors per track (5-1/4 inch drives only).

/B

Leaves room for DOS system files, without actually copying the files (5-1/4 inch drives only).

/N:xx

Specifies the number of sectors per track to format.

/T:yy

Specifies the number of tracks to format.

COMMENTS: You can use the FORMAT command to format diskettes or hard disk drives.

The FORMAT command formats the specified disk, and initializes the root directory and File Allocation Tables (FAT).

CAUTION

The FORMAT command destroys all data on a disk. Use caution before specifying a drive letter for the FORMAT command. On a hard disk, you are prompted with the drive name and a warning message before formatting begins. If a volume label exists for a disk that you want to format, you must supply the current volume label before you are permitted to reformat the disk.

NOTE: Do not use the FORMAT command on disks defined with the ASSIGN, JOIN, or SUBST commands.

Format Compatibility

Before you start formatting diskettes, consider the following compatibility relationships between drive types and the various diskettes:

Drive Type	Diskette Type
160 KB/180 KB	160 KB/180 KB single-sided diskettes
320/360 KB	320 KB/360 KB single-sided or double-sided diskettes
720 KB	720 KB double-sided diskettes[a]
1.2 MB	160 KB/180 KB single-sided[b] 320 KB/360 KB double-sided or high capacity diskettes[b]
1.44 MB	720 KB double-sided diskettes[c] 1.44 MB double-sided diskettes[d]

[a]Do not format 2.0 MB capacity or high-density diskettes in a 720 KB drive.

[b]To format a single-sided or double-sided diskette in a high-capacity drive, use the /4 parameter referred to in this section.

[c]To format a diskette labeled 1.0 MB capacity, 2 HC or with no label, in a 1.44 MB drive, use the FORMAT command with the /N:9 and /T:80 parameters.

[d]To format a diskette labeled 2.0 MB capacity or HD, in a 1.44 MB drive, use the FORMAT command without the /N and /T parameters. No other combinations are allowed.

Options /V, /B, and /S tell FORMAT to take special actions with DOS system-type files on disks as follows:

/S Causes FORMAT to copy the operating system files from the DOS disk in the default drive to the newly formatted disk. The files are copied in the following order: IBMBIO.COM, IBMDOS.COM, and COMMAND.COM. Other files can then be copied selectively using the COPY command.

/V Causes FORMAT to pause in the formatting process and display a message asking you to type in a volume label. The label is a name of up to 11 characters that you choose to identify disks individually.

/B Causes FORMAT to leave room for the DOS operating system without copying it. The resulting diskette is formatted double-sided, 8 sectors per track, and is therefore not used with /V and /8 parameters. This option is not available for high-capacity diskettes.

If the drive contains a diskette designed for lower capacity, or you need to use a nonstandard format, you can use the following options when formatting:

/1 Formats a single side of a diskette only, regardless of the drive type.

/4 Identifies nonhigh-density diskettes in a high-capacity drive.

/8 Formats a diskette using 8 sectors per track instead of the default 9 or 15 sectors per track.

Use options /N:xx and /T:yy when you want to format a diskette to less than the maximum supported capacity of the drive (a 720 KB diskette in a 1.44 MB drive, for example).

Parameter Compatibility

The chart below summarizes the options valid for each type of disk:

Disk Type	Valid Parameters
Single-sided (160/180 KB)	/S, /V, /1, /4, /8, /B
Double-sided (320/360 KB)	/S, /V, /1, /4, /8, /B
Double-sided (720/1440 KB)	/S, /V, /N, /T
High-density (1.2 MB)	/S, /V, /N, /T
Hard disk	/S, /V

FORMAT provides parameters for formatting diskettes for a previous version of MS-DOS. Therefore, certain parameters are incompatible with each other. For example, the /B and /8 options produce 8-sector diskettes, but the /V option is not supported by versions of MS-DOS that support 8-sector diskettes. The /B option is contradictory to the /S option because /B only reserves room for system files on a diskette, while /S copies the system files to the diskette. Also, /B /V /8 is not allowed, while /8 /V /S is allowed.

FREQ

TYPE: Utility

Refer to Appendix D.

GOTO

TYPE: Batch

PURPOSE: To change the sequence of execution of batch file commands.

FORMAT: GOTO label

or

GOTO:label

COMMENTS: GOTO causes batch commands to be executed beginning with the line following the <label> line. The <label> parameter is identified as a colon followed by the label name (see example below). If you do not define a label, the current batch file terminates. The label line can either precede or follow the GOTO line in the batch file.

For example,

```
:abc
REM looping...
GOTO abc
```

produces an infinite sequence of the message:

```
REM looping....
```

GRAFTABL

TYPE: External

PURPOSE: Loads a table of additional character data into memory for a monitor running in color/graphics mode.

FORMAT: [d:][pathname]GRAFTABL [437|860|863|865|/STATUS|?]

WHERE: [d:][pathname]

Indicates the drive and directory path where the GRAFTABL command is located. This option is not necessary if the command is in the current directory or you have previously defined a path to the command.

437

United States (default value)

860

Portugal

863

Canada (Fr.)

865

Norway and Denmark

/STA or STATUS

Displays the number of the selected country code page.

?

Displays a summary of command line options.

COMMENTS: GRAFTABL allows you to display various language character sets while your display adapter is in graphics mode. Very few applications programs run in graphics mode (most run in text mode). You can change language character sets while in text mode by using the CHCP command (described in this section).

GRAPHICS

TYPE: External

PURPOSE: Allows you to print the contents of a graphics display on an acceptable printer.

FORMAT: [d:][pathname]GRAPHICS [COLOR1][COLOR4][COLOR8] [COMPACT][GRAPHICS][/R][/B][LCD]

WHERE: [d:][pathname]

Indicates the drive and directory path where the GRAPHICS command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

COLOR1

Indicates any printer compatible with the IBM PC Color Printer (with black ribbon)

COLOR4

Indicates any printer compatible with the IBM PC Color Printer (with RGB ribbon: red, green, blue, black).

COLOR8

Indicates any printer compatible with the IBM PC Color Printer (with CMY ribbon: cyan, magenta, yellow, black).

COMPACT

Indicates any printer compatible with the IBM PC Compact Printer.

GRAPHICS

Indicates any printer compatible with the standard IBM PC Graphics Printer (default option).

/R

Prints black as black and white as white.

/B

Prints the background color if COLOR4 or COLOR8 is specified.

COMMENTS: To print screen graphics on a printer, type the appropriate GRAPHICS command line, and press the <ENTER> key. You can then print the screen by simultaneously pressing the <Shift> and <PrtSc> keys. The GRAPHICS command is retained by the operating system until you reset the system.

When you use the <PrtSc> key, remember:

- In text mode it takes less than 30 seconds to print the screen; in graphics mode it may take up to 3 minutes.
- In color graphics mode, the screen contents are printed in up to four shades of gray if you specify GRAPHICS or COLOR1.
- If you have 640 pixels across the screen, the screen is printed sideways on the paper.

If you use this command frequently, it may prove helpful to add the command to your AUTOEXEC.BAT file. If you want assembler program to execute GRAPHICS, insert the following in your code:

```
PUSH BP
INT 5
POP BP
```

NOTE: The GRAPHICS command does not enable the printing of characters that are not contained in your printer's character set. The command is for printing graphic screens only.

IF

TYPE: Batch

PURPOSE: Conditionally executes a DOS or batch command during batch file processing.

FORMAT: IF [NOT] condition command

COMMENTS: The IF command first tests <condition>. If <condition> is true, or when modified by the NOT parameter, is false, the <command> is executed. If <condition> is found to be false, or when modified by the NOT parameter, is found to be true, the <command> is not executed.

The <condition> parameter to be tested is one of the following:

ERRORLEVEL n

True only if the previously executed program or command had an exit code of n or higher.

string1==string2

True only if <string1> and <string2> are identical after batch replacement parameter (%%, if used) substitution. Strings cannot have embedded separators. Corresponding characters in each string must be identical, either both uppercase or both lowercase.

EXIST [d:][pathname]filename[.ext]

True only if [d:][pathname]filename[.ext] exists as specified.

NOT condition

True only if the condition is false.

EXAMPLES: In the following example, if the file TMP.BID does not exist, the message "Can't find file" is displayed:

```
IF NOT EXIST TMP.BID ECHO Can't find file
```

In the following example, if the ERRORLEVEL code is not 3, all files on drive B that are named DATA (with any extension) are printed:

```
IF NOT ERRORLEVEL 3 PRINT B:DATA.*
```

JOIN

TYPE: External

PURPOSE: Provides access to a disk drive through a specific directory on a second drive.

FORMAT: To report joins:

[d:][pathname]JOIN

To create joins:

[d:][pathname]JOIN d1: d2:\directory

To delete joins:

[d:][pathname]JOIN d1: /D

WHERE: [d:][pathname]

Indicates the drive and directory path where the JOIN command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1:

Specifies the drive to be joined.

d2:

Specifies the drive to which d1: is joined.

\directory

Indicates the directory name in root of d2 representing d1:.

/D

The "Unjoin" option.

COMMENTS: If you are in drive C:, for example, you can use the JOIN command to:

- Specify an existing *empty* directory in the root directory of drive C:, or create a new directory (C:\DRIVEA, for example).

- JOIN another drive (drive A:, for example) to the directory you specified in drive C:.

  The new directory (C:\DRIVEA) assumes the entire directory structure of drive A:. Once you have joined drive A to directory C:\DRIVEA, you access the directories and files of drive A from directory C:\DRIVEA. You cannot refer to the drive letter A until you "unjoin" drive A by typing the JOIN command line with the /D option.

When you join a drive to a directory, the directory to which you JOIN the drive must be in the root directory. You cannot join a drive to a second level directory, such as C:\TEST\DRIVEA.

The directory to which you join a drive must either not exist yet in the root directory, or must exist and be empty. You cannot join the default drive.

EXAMPLES: To join diskette drive A: to hard disk drive C:, type:

```
JOIN A: C:\DRIVEA
```

and press the <ENTER> key. Drive A: is now joined to C: through directory C:\DRIVEA. A: is no longer recognized as a valid drive specification. You now make reference to the files on A: through directory C:\DRIVEA.

You must not use commands such as BACKUP, RESTORE, FORMAT, DISKCOPY, and DISKCOMP while a join exists. You cannot join network drives or drives that were acted upon by the SUBST or ASSIGN commands.

To display the joins that are currently in effect, type:

```
JOIN
```

and press the <ENTER> key. The following appears:

```
A: =>C:\DRIVEA
```

To "unjoin" A: from C:, type:

```
JOIN A: /D
```

and press the <ENTER> key.

KEYBOARD.SYS

TYPE: Configuration

Refer to the KEYB command description.

KEYB

TYPE: External

PURPOSE: Replaces the keyboard program that is part of the PC ROM BIOS, and allows support of non-U.S. keyboards.

FORMAT:
```
[d:][pathname]KEYB
[xx[,[yyy],[[d:][path]filename[.ext]]]]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the KEYB command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

xx

Specifies the keyboard code you want to use. Refer to Appendix D in the *MS-DOS User's Guide* (Order No. HU94) for country and keyboard codes.

yyy

Specifies the numeric code page that defines the character set. If you omit yyy, the country default page is used.

[d:][path]filename[.ext]

Indicates where the KEYB command should search for the keyboard definition file. If you do not specify a parameter, KEYB looks for the file KEYBOARD.SYS file in the root directory of your current drive.

COMMENTS: NOTE: Keyboard programs from previous versions of MS-DOS are not compatible with MS-DOS version 3.30.

When you load a new keyboard program into memory using the KEYB command, the new program disables the keyboard program resident in the ROM BIOS. The new keyboard program remains in memory until you reset the system, or start it up again.

To return to the United States keyboard format (resident in ROM BIOS), press the <Ctrl>, <Alt>, and <F1> keys simultaneously.

To return to the keyboard program last loaded with KEYB command, press <Ctrl-Alt-F2>.

After using KEYB once to load a new keyboard program, you can use it again to load still another keyboard program. Although the earlier program remains in memory, you can no longer access it. Loading several keyboard programs without performing a system reset results in a considerable loss of memory space. The memory allocated to each keyboard program is approximately 2000 bytes.

The following table shows the acceptable country code (yyy option) and keyboard code (xx option) combinations that you can use with the KEYB command.

Code Page (yyy)	Keyboard Layout (xx)
437	FR, GR, IT, LA, NL, SP, SV, SU, UK, US
850	BE, CF, DK, FR, GR, IT, LA, NL, NO, PO, SF, SG, SP, SU, SV, UK, US
860	PO
863	CF
865	DK, NO

NOTE: You must specify the keyboard code (xx) that corresponds to the keyboard that you are using, or improper characters result from your keystrokes.

When you issue the KEYB command, it creates translation tables for each code page that you have prepared using the MODE command. The requested code page (yyy) becomes the active code page if it was previously prepared. If it has not been prepared, an error message occurs. Refer to the MODE command description for information on preparing code pages. Refer to Appendix C for information on non-U.S. English keyboard formats.

KEYBOARD.SYS The KEYBOARD.SYS file contains tables that direct the KEYB.COM command to convert scan codes to ASCII characters. Refer to the KEYB command description for a listing of valid code page and keyboard combinations, and for information on loading and activating various keyboard tables.

LABEL

TYPE: External

PURPOSE: Creates, changes, or deletes a volume label on a disk.

FORMAT: [d:][pathname]LABEL [d1:][volume label]

WHERE: [d:][pathname]

Indicates the drive and directory path where the LABEL command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1:

Specifies the disk drive containing the disk you want to label.

volume label

A 1- to 11-character volume label (name for a disk).

COMMENTS: Any characters acceptable for file names are acceptable for volume labels. If the volume label exceeds 11 characters, DOS uses only the first 11 characters. If you do not specify a name, you are prompted with the message:

Volume label (11 characters, ENTER for none)?

EXAMPLES: In the following example, a volume label called MYDISK is created on drive C.

```
LABEL C:MYDISK
```

To change this label to YOURDISK, type:

```
LABEL C:
```

The following message is displayed:

```
Volume in drive C is MYDISK

Volume label (11 characters, ENTER for none)?
```

Enter YOURDISK, and press <ENTER>. The following message is displayed:

```
Delete current volume label (Y/N)?
```

Depending on your answer, the volume label remains unchanged or is deleted.

LASTDRIVE

TYPE: Configuration

PURPOSE: Defines the maximum number of disk drives installed in your PC.

FORMAT: LASTDRIVE=x

WHERE: x

Specifies the letter assigned to the highest lettered disk drive. (In DOS, disk drives are labeled A, B,...Z).

COMMENTS: The default value is E, which indicates that access is provided to five drives. If the value you specify is less than the number of physical drives installed in your PC, DOS assumes the default value.

The following example sets the number of drives equal to 12:

```
LASTDRIVE=L
```

The LASTDRIVE command is useful in a network environment, and also when you install memory-resident disks, such as VDISK.

The LASTDRIVE command sets sets the default drive limit used by the SUBST command.

MKDIR (MD)

TYPE: Internal

PURPOSE: Creates a subdirectory on a disk.

FORMAT: MKDIR [d:]pathname

or

MD [d:]pathname

WHERE: [d:]

Specifies the drive in which you are creating the new directory.

pathname

Specifies the pathname of the directory you are creating.

COMMENTS: To create a subdirectory, you must specify the entire pathname of the directory you are creating.

EXAMPLES: If you are in the root directory, the command

```
MKDIR FILES
```

creates a subdirectory FILES in your root directory.

To create a subdirectory named CHUCK under the \FILES directory, type:

```
MD \FILES\CHUCK
```

MODE

TYPE: External

PURPOSE: Defines which printers, serial/parallel communications controllers, or monitor controllers DOS is to use. Selects which code pages are to be used for various devices, and establishes the operational characteristics of each.

FORMAT 1: To select printer characteristics:

```
[d:][pathname]MODE LPT#[:][cpl][,[lpi][,P]]
```

FORMAT 2: To direct output to a serial printer via a serial/parallel communications controller:

```
[d:][pathname]MODE LPT#[:]=COMx
```

FORMAT 3: To specify the operational characteristics of a serial/parallel communications controller:

```
[d:][pathname]MODE COMx[:]baud rate[,[parity]
[,[data bits][,[stop bits][,P]]]]
```

FORMAT 4: To specify the display mode of the color/graphics monitor controller, to shift the screen, or to switch active status between a color/graphics monitor controller and a monochrome monitor controller:

```
[d:][pathname]MODE monitor type
```

or

```
[d:][pathname]MODE [monitor type],shift[,T]
```

FORMAT 5: To prepare code pages:

```
[d:][pathname]MODE device CODEPAGE
PREPARE=((cplist)[d:][path]filename[.ext])
```

or

```
[d:][pathname]MODE device CP
PREP=((cp)[d:][path]filename[.ext])
```

FORMAT 6: To select or activate a code page:

[d:][pathname]MODE device CODEPAGE SELECT=cp

or

[d:][pathname]MODE device CP SEL=cp

FORMAT 7: To display the currently active code page:

[d:][pathname]MODE device CODEPAGE [/STATUS]

or

[d:][pathname]MODE device CP [/STA]

FORMAT 8: To refresh a lost code page:

[d:][pathname]MODE device CODEPAGE REFRESH

or

[d:][pathname]MODE device CP REF

COMMENTS: The MODE command uses eight formats for setting operational characteristics of various devices.

Format 1: Use Format 1 to select printer characteristics. For some printer models, MODE Format 1 sets the printer number, characters per line, lines per inch, and timeout retry parameters. (If a serial printer is installed via a serial communications controller, you must also enter MODE Format 2 to assign the printer output to that device.)

You can have from one to three printers attached to the PC, addressed as LPT1, LPT2, and LPT3. You must identify the first, or primary, printer as LPT1. The reserved device name PRN, discussed in Section 2, always directs DOS to use LPT1 when entered in lieu of a file name for redirecting output (as in DIR | SORT >PRN).

The format for MODE Format 1 is:

```
[d:][pathname]MODE LPT#[:][cpl][,[lpi][,P]]
```

Where:

[d:][pathname]

Indicates the drive and directory path where the MODE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

#

Specifies the printer number (1, 2, or 3).

cpl

Specifies the number of characters per (8-inch wide) line. The number can be either 80 (10 characters per inch) or 132 (16.5 characters per inch). The default value is 80.

lpi

Specifies the number of lines per inch, either 6 or 8. The default value is 6.

P

Causes a continuous retry on printer not ready (timeout) condition.

DOS starts with the printer at 80 characters per line and 6 lines per inch.

This command works best with standard PC printers. If you don't have this type of printer, MODE is not able to set the characters per line or the lines per inch, and displays an error message.

Because MODE also does not accept pathed device names, you cannot use phrases such as \DEV\LPT1.

The [,P] option causes DOS to keep trying to send characters to the printer when something is wrong. To use the P option, you must give the full command with the [,P] every time you use MODE LPTx. The first time you give MODE LPTx: without the [,P] option, DOS does not retry continuously but gives an error message when the printer "times out."

If you use the [,P] option and your printer hangs up, your computer locks up. You can get the computer out of this condition by typing a control-break sequence (<Ctrl-Break> or <Ctrl-C>). You need to wait almost a minute more for DOS to recognize the control-break.

If you don't want to change the characters per line, the lines per inch, or the continuous retries, either drop the option from the command line, or just use the comma and leave out the number.

To change the characters per line to 132 and the lines per inch to 8, type:

```
MODE LPT1:132,8
```

To set the characters per line to 80, but leave the lines per inch the same, type:

```
MODE LPT1:80
```

To leave the characters per line unchanged and change the lines per inch to 6, type:

```
MODE LPT1:,6
```

In all three of the above examples, DOS does not retry on a timeout because you have not included the [,P] option.

The command:

```
MODE LPT1:132,8,P
```

sets the characters per line (cpl) to 132 and lines per inch (lpi) to 8, and specifies continuous retry on timeout.

If you later enter:

```
MODE LPT1:,6
```

the cpl is unchanged (at 132), the lpi changes to 6, and continuous retry on timeout is stopped.

Format 2: Format 2 directs printer output to a serial printer via a serial communications controller. The format is:

[d:][pathname]MODE LPT#:=COMx

Where:

[d:][pathname]

Indicates the drive and directory path where the MODE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

#

Specifies the number of the printer you are reassigning.

x

Specifies the communications controller for your serial printer.

This command directs DOS to use the serial printer connected to a communications controller instead of the normal parallel printer. The communications controller number can be either 1, 2, 3 or 4.

The following command:

MODE LPT1:=COM1

redirects DOS to use the printer that is connected to the first communications controller.

To reverse this setting, use the MODE LPT1 command.

To use this version of MODE, you must first set up the serial communications controller using format 3 of the MODE command.

Format 3: MODE Format 3 sets the serial/parallel communications controller characteristics: baud rate, parity, data bits, stop bits, and retries. The format is:

```
[d:][pathname]MODE COMx[:]baud rate
[,[parity][,[data bits][,[stop bits][,P]]]]
```

Where:

[d:][pathname]

Indicates the drive and directory path where the MODE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

x

Specifies the number of the serial/parallel communications controller (either 1, 2, 3 or 4).

baud rate

110, 150, 300, 600, 1200, 2400, 4800, 9600 or 19200. (You can abbreviate and use just the first two numbers of the rate.)

parity

O, E, or N for Odd, Even, or None. The default value is Even.

data bits

7 or 8. The default value is 7.

stop bits

Either 1 or 2. If you use 110 bps, 2 stop bits is the default. Any baud rate other than 110 defaults to 1 stop bit.

P

"Keep trying."

<baud rate> is the only required parameter. All other parameters are optional. When you include the optional parameters, however, they must be in the sequence shown in the format line. Use a comma to indicate an omitted parameter.

If you do not set the P option, it is OFF. (Use this option with a serial printer.) Use Retry for printers, but avoid it when the controller is being used with a modem. Use a <Ctrl-Break> or <Ctrl-C> to get out of a timeout retry loop.

Either one of the following two command lines

```
MODE COM1: 1200
```

or

```
MODE COM1: 12
```

set the baud rate at 1200 for the first communications controller. Everything but retry (,P) remains the same. (Even if retry was ON, it is now OFF because you have not included the P option.)

The command line

```
MODE COM1: 96,,8
```

sets the first controller to 9600 bps (or bits per second) and 8 data bits. Everything else but the retry is unchanged. Retry is turned off.

The command line

```
MODE COM1: 48,,,,P
```

sets the controller to 4800 bps and continuous retries. As before, nothing else is changed.

Format 4: Format 4 handles the monitors you have connected to your color graphics monitor board. If you have only the monochrome monitor, this command does nothing.

The format is:

[d:][pathname]MODE monitor type

or

[d:][pathname]MODE [monitor type],shift[,T]

Where:

[d:][pathname]

Indicates the drive and directory path where the MODE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

monitor type

40, 80, BW40, BW80, CO40, CO80, or MONO.

shift

Shifts the screen L(left) or R(right).

T

Requests a Test pattern.

The monitor type can be any one of the following (The C/G monitor refers to the RGB monitor, television monitor, or composite monitor attached to the color/graphics monitor controller board):

40	40-column lines for the screen.
80	80-column lines for the screen.
BW40	Makes the C/G monitor the active monitor, uses 40-character lines, and turns off color for the composite monitor.
BW80	Makes the C/G monitor the active monitor, uses 80-character lines, and turns off color for the composite monitor.
CO40	Makes the C/G monitor the active monitor, turns on color capabilities, and uses 40-character lines.
CO80	Makes the C/G monitor the active monitor, turns on color capabilities, and uses 80-character lines.
MONO	Makes the high-resolution monochrome monitor the active monitor.

The active monitor is the current monitor being used by DOS and your programs to display information. If you have two monitors, such as a high-resolution monochrome monitor and another monitor or TV attached to the C/G board, either one can be used with this command.

Color is optional. Some programs use color; others do not. Specifying CO40 or CO80 does not guarantee that your programs will display in color. The selection merely lets DOS display output in color. It's up to you and your programs to make color appear.

MODE can only partially be overridden by BASIC. If you use a color command in BASIC, it makes no difference whether you specify BW40, BW80, CO40, or CO80. BASIC still tries to send colors. Whether you see colors or not depends on the monitor or TV you use. However, the size of the screen (40 or 80 columns) and the active monitor selected by the MODE command remains in effect until you change it with another MODE command.

The shift option (R or L), shifts the screen left or right one character position. You can use shift with either a television set or a composite monitor to adjust the screen. With some composite monitors, characters "fall off" the edge of the screen. This command lets you shift the line back into position.

The T option produces a test pattern line of either 40 or 80 characters. After displaying the test pattern, MODE asks whether the screen is okay. Enter Y if it is. If you enter N, MODE shifts the screen by one character and repeats the question. In this way, you can adjust your screen without having to reenter the MODE command.

You should enter both the S and T options when the prompt is on the active monitor that you want to adjust.

For example

```
MODE 40,L,T
```

sets the mode to 40 characters per line, and shifts the entire screen one character position to the left. A test pattern is displayed that essentially counts the character positions on the line. You can then shift the screen further to the left without reentering the MODE command.

Format 5: Use Option 5 to prepare code pages. The format is:

```
[d:][pathname]MODE device CODEPAGE
PREPARE=((cplist)[d:][pathname]filename[.ext])

or

[d:][pathname]MODE device CP
PREP=((cp)[d:][pathname]filename[.ext])
```

Where:

[d:][pathname]

Indicates the drive and directory path where the MODE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

device

Specifies one of CON, PRN, LPT1, LPT2, or LPT3.

cp

Specifies one code page number.

cplist

Specifies a list of code pages. The code page must be of the following values: 437, 850, 860, 863, or 865. If cplist is a list of code pages, the code page list must be enclosed in (and).

[d:][pathname]filename[.ext]

Specifies the location and name of the file containing the code pages. The code page information files located on the DOS diskette have the extension of .CPI.

```
4201.CPI = IBM Proprinter
5202.CPI = IBM Quietwriter III
EGA.CPI  = EGA type devices
```

Refer to Appendix D in the *MS-DOS User's Guide* (Order No. HU94) for more information on code page switching.

Format 6: Use Option 6 to select or activate code pages. The format is:

[d:][pathname]MODE device CP SEL=CP

Where:

[d:][pathname]

Indicates the drive and directory path where the MODE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

device

Specifies one of CON, PRN, LPT1, LPT2, or LPT3.

cp

Specifies the code page to be activated. The supported code pages are, 437, 850, 860, 863, or 865. The code page to be activated must be located in the list prepared by Format 5.

The device name CON LPT# is set to the specified code page.

Refer to Appendix D in the *MS-DOS User's Guide* (Order No. HU94) for more information on code page switching.

Format 7: Use Option 7 to display the currently active code page. The format is:

[d:][pathname]MODE device CODEPAGE [/STATUS]

or

[d:][pathname]MODE device CP [/STA]

Where:

[d:][pathname]

Indicates the drive and directory path where the MODE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

device

Specifies one of CON, PRN, LPT1, LPT2, or LPT3.

The active code page and a list of selectable code pages for an active device such as CON or LPT# is displayed.

The list of code pages is divided into hardware and prepared code page sections. The hardware section contains code pages defined as hwcp in the device command. The prepared code page section contains code pages prepared through a MODE command.

Format 8: Use Option 8 to refresh a lost code page. The format is:

[d:][pathname]MODE device CODEPAGE REFRESH

or

[d:][pathname]MODE device CP REF

Where:

[d:][pathname]

Indicates the drive and directory path where the MODE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

device

Specifies one of CON, PRN, LPT1, LPT2, or LPT3.

This command reestablishes the active code page if it has been lost. Code pages can be lost in different ways. One way is when you turn off the printer. After turning the printer off and on, a printer may have a different active code page. then the one in the printer driver. It is then necessary to do a refresh command to get back the original code page.

Refer to Appendix D in the *MS-DOS User's Guide* (Order No. HU94) for more information on code page switching.

NOTE: A MODE command in Format 5 must be issued before a MODE command in Format 8.

MORE

TYPE: External

PURPOSE: Displays output one screen at a time, allowing you to see the displayed data before continuing.

FORMAT: [d:][pathname]MORE

WHERE: [d:][pathname]

Indicates the drive and directory path where the MORE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

COMMENTS: MORE is a filter that reads from standard input (such as a command from your terminal) and displays one screen of data at a time. The MORE command then pauses and displays the "-- MORE --" message at the bottom of the screen.

Press any key to display another screen of data. This process continues until all the input data has been read. To stop the execution of the command, press <Ctrl-Break> or <Ctrl-C>.

The MORE command is useful for viewing a long file or directory one screen at a time. If you type:

```
MORE < MYFILES.TXT
```

or

```
TYPE MYFILES.TXT | MORE
```

DOS displays the file MYFILES.TXT (from the default drive) one screen at a time.

MORE writes a temporary file on the default disk drive, and uses this file as a buffer during the display operation. Therefore, if the disk in the default drive is either "write-protected" or is full, the MORE command cannot function.

NLSFUNC

TYPE: Internal External

PURPOSE: Provides support for extended country information. Allows you to use the CHCP command to select code pages for all devices with code page switching support. You must load NLSFUNC before using the CHCP command.

FORMAT:
```
[d:][pathname]NLSFUNC
[[d:][pathname]filename[.ext]]
```

WHERE: `[d:][pathname]`

Indicates the drive and directory path where the NLSFUNC command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

`[d:][pathname]filename`

Specifies the drive and directory path in which the country information file is located, and the name of the file (COUNTRY.SYS). If you do not use this parameter, the drive path and file name defined by the COUNTRY= command in the CONFIG.SYS file are used. You must include the file name if you specify the drive or path.

EXAMPLE: The following example loads the NLSFUNC command, and specifies that the COUNTRY.SYS file is located in a directory called SYSFILES:

```
NLSFUNC C:\SYSFILES\COUNTRY.SYS
```

PARK

TYPE: Utility

Refer to Appendix D.

PATH

TYPE: Internal

PURPOSE: Establishes the directory path that DOS is to follow when looking for an external DOS command, user command, or application command. PATH only works for external commands ending in .EXE, .COM, or .BAT.

NOTE: You can create a directory path for nonexecutable files (ending in extensions other than .EXE, .COM, or .BAT) by using the APPEND command. Refer to the APPEND command description for information.

FORMAT: PATH [[d1:]pathname1[[;[d2:]pathname2]...]]

or

PATH ;

WHERE: [dn:][pathname]

Specifies each of the drive and directory segments associated with the path that you want DOS to follow.

;

Is the delimiter used between pathnames.

COMMENTS: The PATH command allows you to define which directories DOS searches for an external command after it searches your working directory, and does not find the command.

EXAMPLES: To tell DOS to search your CHUCK directory on drive C after it has searched the current directory for the external commands, type:

PATH C:\USER\CHUCK

and press the <ENTER> key.

DOS now searches the current directory or the default drive, then the CHUCK directory on drive C for external commands, until you set another path.

You can tell DOS to search more than one directory by specifying several pathnames separated by semicolons. For example,

```
PATH C:\USER\CHUCK;B:\USER\BILL;C:\DEV
```

tells DOS to search for external commands on the drives and in the directories specified by the above pathnames. DOS searches the pathnames in the order specified in the PATH command.

If you type the PATH command without any options, the current path listing is displayed. If you specify the following command, DOS sets the NUL path (only the current directory is searched for external commands):

```
PATH ;
```

If you want to execute a command or program that is located in a directory that is not contained in the path listing, you can specify the pathname for the command or program <d:pathname> as part of the command line. For example, if you want to use the DEBUG command, but the directory in which the DEBUG.COM file is located (C:\PROGRAM, for example) is not in the PATH listing, you can type the DEBUG command as follows:

```
C:\PROGRAM\DEBUG
```

PAUSE

TYPE: Batch

PURPOSE: Suspends the execution of batch commands and displays a message.

FORMAT: PAUSE [message]

COMMENTS: During the execution of batch commands, you may need to change disks or perform some other action. PAUSE suspends the execution of the batch command, prints a specified message (if you choose to type one), and waits for you to press a key:

```
Pause <message>
Strike a key when ready . . .
```

When you press any key, the batch process resumes execution.

If you press <Ctrl-Break> during the execution of a batch file, the following prompt appears:

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

If you type Y in response to this prompt, execution of the remainder of the batch file is aborted, and control is returned to the operating system command level. Therefore, you can use PAUSE to break a batch file into segments, allowing the option of ending the batch processing at an intermediate point.

The <message> is optional. If you want to display a message, type it on the same line as PAUSE. You may want to prompt the user with a relevant message, for example, when the batch file pauses.

The <message> is displayed only if ECHO is ON.

PRINT

TYPE: External

PURPOSE: Prints a text file on a printer while you are processing other DOS commands (usually called "background printing").

FORMAT:
```
[d:][pathname]PRINT
[/D:device][/B:buffsize][/U:busyticks]
[/M:maxticks]
[/S:timeslice][/Q:queuesize][/T][/C][/P]
[[d1:][pathname1]filename1[.ext1]...]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the PRINT command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

[d1:][pathname1]

Specifies the drive and directory path of the print file.

filename1[.ext1]

Specifies the file name and extension of the print file.

/D:

Device type

/B:

Buffer size

/U:

Busy ticks

/M:

Max ticks

/S:

Time slice

/Q:

Print queue size

/T

Terminate

/C

Cancel

/P

Print

COMMENTS: You can only use the PRINT command if you have a printer attached to your PC.

You can only use the options /D, /B, /U, /M, /S, and /Q the first time you run the PRINT command after loading DOS. Specifying them again results in an "Invalid Parameter" error.

Following are descriptions of the PRINT command options:

/D:

(device) Specifies the print device (LPT1, LPT2, LPT3, PRN, COM1, COM2, COM3, COM4, AUX, etc.). The default print device is PRN. If you do not use this option, DOS prompts you for the name of the print device.

/B:

(buffer size) Determines the size in bytes of the internal buffer (from 512 to 16384 bytes). The default is 512 bytes. To enhance the performance of the PRINT command, increase the value of /B.

/U:

(busy ticks) Specifies the number of clock ticks that PRINT waits until the printer is available. PRINT gives up its timeslice if it waits longer than the value of /U. If you do not specify this value, PRINT assumes the value 1.

/M:

(max ticks) Specifies how many clock ticks PRINT can have to print characters (from 1 to 255 clock ticks). The default value is 2 clock ticks.

/S:

(timeslice) Determines the timeslice value (from 1 to 255 time slices). The default is 8.

/Q:

(queue size) Specifies the number of files allowed in the print queue (from 4 to 32). The default value is 10.

/T

(terminate) Deletes all files in the print queue (the file currently being printed, and those waiting to be printed). A message to this effect is printed. The alarm, if any, is sounded, and the paper is advanced to the top of the next page.

/C

(cancel) Turns on cancel mode. The preceding file name and all following file names are suspended in the print queue until you type a /P option on the command line, or until you press <ENTER>.

/P

(print) Turns on print mode. The preceding file name and all following file names are added to the print queue until you issue a /C option on the command line, or until you press <ENTER>.

If you type PRINT with no options, the contents of the print queue appear on your screen without affecting the queue.

NOTE: A diskette containing the files to be printed cannot be removed during printing. Neither the files in the print queue nor the printer can be used in normal processing until all printing is complete.

If a disk error occurs, DOS displays a message to this effect. The file currently being printed is canceled, the paper is advanced to the bottom of the page and the alarm, if any, is sounded. Any files still in the print queue are printed.

EXAMPLES: To empty the print queue, type:

```
PRINT /T
```

To empty the print queue and queue all .ASM files on the default drive, type:

```
PRINT /T*.ASM
```

To remove the three specified files from the print queue, type:

```
PRINT A:TEMP1.TST/C A:TEMP2.TST A:TEMP3.TST
```

To remove TEMP1.TST from the queue, and add TEMP2.TST and TEMP3.TST to the queue, type:

```
PRINT TEMP1.TST/C TEMP2.TST /P TEMP3.TST
```

PRINTER.SYS

TYPE: Configuration

Refer to the DEVICE command description.

PROMPT

TYPE: Internal

PURPOSE: Changes the DOS command prompt, or inputs escape sequences.

FORMAT: PROMPT [prompt-text]

COMMENTS: To change the text of the DOS system prompt, type the PROMPT command followed by the text that you want displayed. For example, type the following command to set the DOS prompt to HELLO:

```
PROMPT HELLO
```

You can use any displaying keyboard characters in the prompt, except the following:

```
$  >  <  |  =
```

You can also use a set of special codes for use in prompts in addition to text (as shown in the table of PROMPT codes on the following page). You can use these codes in any combination. For example, the following command sets the prompt to the current drive letter and pathname followed by >:

```
PROMPT $P$G
```

The special codes can also be intermixed with standard characters to build a prompt line. For example:

```
PROMPT $N:PJG
```

sets the prompt to the default drive letter, followed by a colon and the letters PJG (which could be a set of initials).

PROMPT CODES

Use This Code:	To Get This Prompt:
$t	The current time in the form "hh:mm:ss"
$d	The current date in the form "day-of-week mm-dd-yy"
$p	The current drive and directory in the form "d:\current path"
$v	The version number in the form "MS-DOS Version X.XX"
$n	The default drive letter
$$	The "$" character
$g	The ">" character
$l	The "<" character
$b	The "|" character
$q	The "=" character
$s	A space
$_	A carriage return/line feed sequence (<ENTER>)
$h	A back space, erasing the previous character
$e	ASCII code X'1B' (ESCape)

EXAMPLES: The following example sets the DOS prompt to display the time on one line and the date on the next:

```
PROMPT Time = $T$ Date = $D
```

The DOS prompt appears as follows, updating the date and time each time it is redisplayed:

```
Time = 16:16:37.09
Date = Wed  6-23-1988
```

You type commands on the second line after the date.

You cannot begin prompt-text with one of the DOS command delimiters:

```
,   ;   =
```

(or space or tab). To create a prompt beginning with one of the DOS command delimiters, use the dollar sign ($) followed by a character not defined in the table of PROMPT codes to begin the prompt-text. For example:

```
PROMPT $A;ABC
```

Since the code $a is not included in the table, DOS interprets A as a null character and uses the remainder of the line as the DOS prompt.

If you enter the PROMPT command without prompt-text, the prompt is reset to the default DOS system prompt (the default drive designation followed by the > character).

You can also use the PROMPT command with escape functions beginning with $e[to program your function keys, the keyboard, the cursor, and display graphics. Refer to Section 4 for details.

RAMDRIVE.SYS

TYPE: Configuration

Refer to the DEVICE command description.

RECOVER

TYPE: External

PURPOSE: Recovers a file or an entire disk containing bad sectors or a bad directory.

FORMAT:
```
[d:][pathname]RECOVER
[d1:][pathname1]filename1[.ext1]

or

[d:][pathname]RECOVER d1:
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the RECOVER command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

[d1:][pathname1]

Specifies the drive and directory path of the file to be recovered.

filename1[.ext1]

Specifies the file name and extension of the file to be recovered.

COMMENTS: If sectors on a disk are bad, you can recover either the file that contains the bad sectors (without the bad data in the sector, but saving the rest of the file) or the entire disk (if the bad sector was not in the directory).

To recover a particular file, type:

```
RECOVER filename
```

DOS responds with:

```
Press any key to begin recovery of the
file(s) on drive=h
```

DOS reads the file sector by sector and skips the bad sector(s).

To recover a disk, type:

```
RECOVER d:
```

where d: is the letter of the drive containing the disk to be recovered.

When you recover an entire disk, the directory is rebuilt, and all files are renamed FILEnnnn.REC, where <nnnn> is a four-digit number beginning with 0000. Subdirectories are also renamed, but not rebuilt, in the FILEnnnn.REC format and placed in the root directory.

If there is not enough room in the root directory, RECOVER prints a message, and stores information about the extra files in the File Allocation Table. You can run RECOVER again to regain these files when there is more room in the root directory.

REM

TYPE: Batch

PURPOSE: Displays the message (during batch processing) that you type on the same line as the REM command.

FORMAT: REM [message]

COMMENTS: The only separators allowed in the <message> are the space, tab, and comma. The maximum length of the message is 123 bytes. Following is an example of a batch file using REM commands:

```
REM This file checks new disks

REM It is named NEWDISK.BAT

PAUSE Insert new disk in drive B:

FORMAT B:/S

DIR B:

CHKDSK B:
```

You can include REM commands with no messages in your batch files to create spacing between commands, making it easier to read the file when it is displayed or printed.

The REM command and its associated message is not displayed during batch file processing if ECHO is OFF.

RENAME (REN)

TYPE: Internal

PURPOSE: Changes the name of a file.

FORMAT:
```
RENAME [d:][pathname]filename1[.ext1]
filename2[.ext2]

or

REN [d:][pathname]filename1[.ext1] filename2[.ext2]
```

WHERE: [d:][pathname]

Specifies the drive and directory path of the file to be renamed.

filename1[.ext1]

Specifies the filename and extension of the file(s) to be renamed.

filename2[.ext2]

Specifies the filename and extension of the new name of the file(s).

COMMENTS: You must give filename1 a drive designation if the disk resides in a drive other than the default drive. The filename2 remains in the same path and directory and on the same disk where filename1 currently resides.

You can use the wildcard characters * or ? in either option. All files matching filename1 are renamed.

For example, the following command changes the names of all files with the .LST extension to the same names with a .PRN extension:

```
REN *.LST *.PRN
```

In the next example, REN renames the file SAM on drive B: to TED:

```
REN B:SAM TED
```

The file remains on drive B.

If you attempt to rename a file to a name already present in the directory, the error message "Duplicate filename or file not found" is displayed.

REPLACE

TYPE: External

PURPOSE: Updates previous versions of files.

FORMAT:
```
[d:][pathname]REPLACE
[d1:][pathname1]filename[.ext] [d2:][pathname2]
[/A][/D][/P][/R][/S][/W]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the REPLACE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

[d1:][pathname1]filename[.ext]

Specifies the disk drive, path, and file name of the new file versions.

[d2:][pathname2]

Indicates the drive and directory path of the old file(s) to be replaced.

/A

Adds new files to the target directory instead of replacing existing ones. You cannot use this option with either the /D or /S options.

/D

Replaces files in the target directory only if the source files are newer than the corresponding target files. This option is incompatible with the /A option.

/P

Prompts you before replacing a target file or adding a source file with the message:

Replace (filename) ? (Y/N)

/R

Replaces read-only files as well as unprotected files. If you do not specify this option, any attempt to replace a read-only file causes an error and stops the replace process. (If a nonread-only file replaces a read-only file, it acquires the read-only attribute.)

/S

Causes REPLACE to search all subdirectories of the target directory while it replaces matching files. This option is incompatible with the /A option. Replace never searches subdirectories in the source path.

/W

Waits for the user to press any key before replacing files. If you do not specify this option, REPLACE begins replacing or adding files immediately.

As files are replaced or added, REPLACE displays the file name on the screen. At the conclusion of the replace operation, it displays a summary line:

NNN file(s) added/replaced

or

No files added/replaced

You cannot use the REPLACE command to update hidden files or system files.

EXAMPLES: Replacing Files:

If your hard disk (drive C) contains several files of client names and phone numbers, you can replace these files with more recent versions that exist on the disk in drive A. For example, type:

```
REPLACE A:\PHONES.CLI C:\/S
```

to replace every file on drive C that is named PHONES.CLI with the file PHONES.CLI from the root directory of drive A.

Adding Files:

To add new printer device drivers to a directory called C:\MSTOOLS, which already contains several printer driver files for a word processor, type:

```
REPLACE A:*.PRD C:\MSTOOLS /A
```

This command adds any files from the default directory of drive A with an extension of PRD (that do not currently exist in the \MSTOOLS directory on drive C) to C:\MSTOOLS.

Exit Codes:

The exit code for this command is one of the following:

0 NORMAL COMPLETION

2 FILE NOT FOUND

3 PATH NOT FOUND

5 ACCESS DENIED (retry using /R parameter)

8 INSUFFICIENT MEMORY

11 INVALID COMMAND LINE

15 INVALID DRIVE

22 UNKNOWN COMMAND (invalid DOS version)

You can test the returned exit code with the IF ERRORLEVEL command in a batch file. Refer to the IF command description, earlier in this section.

RESTORE

TYPE: External

PURPOSE: Restores files created by the BACKUP command.

FORMAT:
```
[d:][pathname]RESTORE d1:
[d2:][pathname2][filename2[.ext2]][/S]
[/P][/B:date][/A:date][/E:time][/L:time][/M][/N]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the RESTORE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1:

Specifies the disk drive containing the backup files.

[d2:][pathname2]

Specifies the drive and directory path of the file(s) to be restored.

filename2[.ext2]

Specifies the file name and extension of the file(s) to be restored (you can use the wildcard characters, * and ?).

/S

Restores the files in the specified directory plus its subdirectories, if any exist. With this option the directory tree structure, which existed when the files were backed up, is re-created and added to your current directory before the files are RESTORED. If you do not use this option, subdirectory files are ignored.

/P

Supplies prompts on the screen that allow you to determine if you want to restore either: files that have changed size since they were last backed up, or read-only files. (If a file is split between two diskettes during the backup process, it is marked as modified even if it remains unchanged.)

/B:

Restores files that were last modified on or before the given date. Refer to the BACKUP command description for the proper date format.

/A:

Restores files that were last modified on or after the given date. Refer to the BACKUP command description for the proper date format.

/E:

Restores files that were last modified at or earlier than the given time. Refer to the BACKUP command description for the proper time format.

/L:

Restores files that were last modified at or later than the given time. Refer to the BACKUP command description for the proper time format.

/M

Restores files that have been modified since the last backup.

/N

Restores files that no longer exist on the target disk.

RESTORE

COMMENTS: When you have used the BACKUP command to store files on a disk, use the RESTORE command to copy those files onto another disk. This command only restores files that you saved using the BACKUP command. (Refer to the BACKUP command description.)

Things to remember when using RESTORE:

- The source of the RESTORE command must be a hard disk or one or more diskettes containing files created through the BACKUP command.
- If the backup files to be restored are on several diskettes, the RESTORE command prompts you to insert the next diskette until all the files have been restored.
- If you do not use the /S option, only the current directory is restored.
- When restoring files to a directory other than the current directory, the target path must originate from the root directory.
- Backup files can be restored only to a directory that has the same pathname as the pathname the file was backed up under.

EXAMPLES: To restore the file called OLDFILE.DAT from the diskette in drive A to the current directory of drive C, type:

```
RESTORE A: C:OLDFILE.DAT
```

and press the <ENTER> key. This command fails if OLDFILE.DAT did not originate from a directory with the same name as your current directory. If OLDFILE.DAT is not found on the diskette in drive A, RESTORE prompts you to insert the next sequential BACKUP diskette.

To restore all files on the backup diskettes in drive A to drive C, including all subdirectory files, type:

```
RESTORE A: C: /S /P
```

Prompts appear that allow you to determine if you want to restore either files that have changed size since they were last backed up or read-only files.

To restore all files ending in .DAT, to the current directory, type:

```
RESTORE A: C:*.DAT
```

To restore all files from diskette A into the LEVEL2 directory on drive C, type:

```
RESTORE A: C:\LEVEL2 /P
```

Prompts appear that allow you to determine if you want to restore either files that have changed size since they were last backed up or read-only files.

If the current directory is not LEVEL3, the following command restores OLDFILE.DAT from drive A to the LEVEL3 directory on drive C:

```
RESTORE A: C:\LEVEL2\LEVEL3\OLDFILE.DAT
```

If the current directory is LEVEL3, use the following command:

```
RESTORE A: C:OLDFILE.DAT.
```

To restore all files in the current directory that were created on or before 6:00 PM, type:

```
RESTORE A: C: /T:6:00P
```

Exit Codes The exit code for this command is one of the following:

0 = NORMAL COMPLETION

1 = NO FILES FOUND TO RESTORE

3 = TERMINATED BY THE OPERATOR
(through <Ctrl-Brk> or <Esc>)

4 = TERMINATED DUE TO ERROR

You can test the returned exit codes with the IF command ERRORLEVEL number condition parameter in a batch file. (Refer to the IF command description in this section.)

You may also want to set VERIFY to ON to check if there are any errors while writing to the disk. (Refer to the VERIFY command description in this section.)

NOTE: If an ASSIGN, JOIN, or SUBST command was in effect when the files were backed up, the files cannot be restored.

RMDIR (RD)

TYPE: Internal

PURPOSE: Removes an empty directory or subdirectory.

FORMAT: RMDIR [d:]pathname

or

RD [d:]pathname

COMMENTS: This command removes a (sub)directory that is empty of files (except for the . and .. entries).

To remove the \USER\CHUCK directory, first issue a DIR command to ensure that the directory does not contain any important files that you do not want deleted. If files that can be deleted exist, use DELETE or ERASE to delete all except the . and .. entries in the (sub)directory. Then, from the root directory, type:

```
RMDIR \USER\CHUCK
```

The CHUCK (sub)directory is deleted from the directory structure.

NOTE: Use care when JOIN or ASSIGN are in effect. You cannot remove a directory that has been used in a SUBST (substitute).

SELECT

TYPE: External

PURPOSE: Installs DOS on a new disk with the selected keyboard layout, date, time, numeric delimiter, and currency formats.

FORMAT: [d:][pathname]SELECT [A:|B:] [d2:][pathname2] [ccc] [kk]

WHERE: [d:][pathname]

Indicates the drive and directory path where the SELECT command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

[A:|B:]

Specifies the source drive. Only A: and B: are valid source drives (A: is the default).

[d2:][pathname2]

Specifies the target drive (B: is the default).

ccc

Specifies the country code.

kk

Specifies the keyboard code.

COMMENTS: Use the SELECT command to install DOS on a disk with the keyboard and date/time options selected. The disk is set up to use the international keyboard layout, date, time, number delimiters, and currency format you choose. The keyboard layout is determined by the keyboard code, and the date, time, number, and currency formats by the country code you select. Refer to the *MS-DOS User's Guide* (Order No. HU94) for more information on this command.

Following are the country and keyboard codes supported by DOS:

Country	Keyboard Code	Country Code	Date Format	Time Format
UNITED STATES	US	001	M-D-Y	12H hh:mm:ss
CANADIAN-FRENCH	none	002	Y-M-D	24H hh:mm:ss
LATIN AMERICA		003	D/M/Y	hh:mm:ss
NETHERLANDS	none	031	D-M-Y	24H hh:mm:ss
BELGIUM	none	032	D/M/Y	24H hh:mm:ss
FRANCE	FR	033	D/M/Y	24H hh:mm:ss
SPAIN	SP	034	D/M/Y	24H hh:mm:ss
ITALY	IT	039	D/M/Y	24H hh:mm:ss
SWITZERLAND	none	041	D.M.Y	24H hh.mm.ss
ENGLAND	UK	044	D-M-Y	24H hh:mm:ss
DENMARK	none	045	D-M-Y	24H hh.mm.ss
SWEDEN	none	046	Y-M-D	24H hh.mm.ss
NORWAY	none	047	D/M/Y	24H hh.mm.ss
GERMANY	GR	049	D.M.Y	24H hh.mm.ss
AUSTRALIA	none	061	D-M-Y	24H hh:mm:ss
PORTGUAL	none	351	D/M/Y	24H hh:mm:ss
FINLAND	none	358	D.M.Y	24H hh:mm:ss
MIDDLE EAST	none	785	D/M/Y	12H hh.mm.ss
ISRAEL	none	972	D M Y	24H hh:mm:ss

The backup copy contains an AUTOEXEC.BAT file with a KEYBkk command, where kk is the keyboard program you selected; and a CONFIG.SYS file with a COUNTRY=ccc command, where ccc is the country code you selected.

SET

TYPE: Internal

PURPOSE: Sets one string equal to another string for access by programs and commands.

FORMAT: SET [string1=[string2]]

WHERE: string1

Indicates the actual string you are defining.

string2

Specifies the parameter you want string1 to equal.

COMMENTS: This command is meaningful only if you want to define (set) values that are used by programs you have written, or by DOS commands.

When you enter the SET command without an accompanying string parameter the current SET values are displayed. These values typically consist of:

- PATH = pathnames
- COMSPEC = A:\COMMAND.COM (the location of your command processor)
- PROMPT = for any prompt variations.

The following example adds the string TTY=VT52 to the environment:

```
SET TTY=VT52
```

If you use SET with only string1, you remove the previous assignment. For example, to remove TTY=VT52 from the environment, type:

```
SET TTY=
```

You can also use the SET command in batch processing. In this way, you can define your replacement parameters with names instead of numbers. If your batch file contains the statement LINK %FILE%, you can set the name that DOS uses for that variable with the SET command. The command SET FILE=DOMORE replaces the %FILE% parameter with the file name DOMORE. Therefore, you do not need to edit each batch file to change the replaceable parameter names. When you use text (instead of numbers) as replaceable parameters, the name must end with a percent sign.

SHARE

TYPE: External

PURPOSE: Installs file sharing and locking.

FORMAT: [d:][pathname]SHARE [/F:ssss][/L:ll]

WHERE: [d:][pathname]

Indicates the drive and directory path where the SHARE command is located. This option is not necessary if the command is in the current directory or you have previously defined a path to the command.

/F:ssss

Allocates file space (in bytes) for the area DOS uses to record file sharing information. Each file that is open needs the length of the full file name plus 11 bytes (the average pathname is 20 bytes). The default value is 2048 bytes.

L:ll

Allocates the number of locks you want to allow. The default value is 20 locks.

COMMENTS: The SHARE command loads file sharing support for networking. Once the SHARE command has been executed, it remains in effect until the system is restarted.

Once you have used the SHARE command in a DOS session, all read and write requests are checked by DOS against the file sharing code.

The message "Share Already Installed" is displayed if you try to load file sharing when it has already been installed.

SHELL

TYPE: Configuration

PURPOSE: Loads a specified command processor file instead of COMMAND.COM.

FORMAT: SHELL [d:][pathname]filename[.ext][/E:eeeee][/P]

WHERE: [d:][pathname]

Specifies the drive and directory path of the new command processor file.

filename[.ext]

Specifies the file name and extension of the new command processor file.

/E:eeeee

Specifies the (base 10) number of bytes reserved for environment strings for the new command processor.

/P

Causes COMMAND.COM to remain loaded and to execute the AUTOEXEC.BAT startup file as soon as the command processor is installed.

COMMENTS: The SHELL command does not affect the value held in teh COMSPEC= environment string. Refer to the SET command description for information on COMSPEC.

SHIFT

TYPE: Batch

PURPOSE: Allows access to more than 10 replacement parameters in batch file processing by shifting all parameters down one parameter position.

FORMAT: SHIFT

COMMENTS: Normally, command files are limited to handling 10 variable parameters, %0 through %9. To access more than 10 parameters, use SHIFT to move the command line parameters down one position. For example, if you enter:

```
BASICOMP A:PROG1 B:PROG2
%0 = BASICOMP
%1 = A:PROG1
%2 = B:PROG2
%3...%9 are empty
```

a SHIFT command results in the following:

```
%0 = A:PROG1
%1 = B:PROG2
%2...%9 are empty
```

If you enter more than 10 parameters on a command line, those that appear after the 10th (%9) are shifted one at a time into %9 by successive shifts (%9 is shifted to %8 and so on, with 0% being replaced by %1).

SMARTDRV.SYS

TYPE: Configuration

Refer to the DEVICE command description.

SORT

TYPE: External

PURPOSE: Reads input from a file or your terminal, sorts the data (according to the ASCII collating sequence), then writes the data to an output device.

FORMAT: [d:][pathname]SORT [/R][/+n]

WHERE: [d:][pathname]

Indicates the drive and directory path where the SORT command is located. This option is not necessary if the command is in the current directory or you have previously defined a path to the command.

/R

Requests a reverse order of sort.

/+n

Specifies the starting column of the sort.

COMMENTS: The SORT command arranges the sequence of records or lines in the input on the terminal or in a file according to ASCII sequence (normally ascending). The sort begins in the specified column (either column 1 or /+n), and includes all characters in the record from the specified column. There are two options that you are allowed to select:

/R

Reverses the sort sequence from ascending to descending order (sorts from Z to A).

/+n

Sorts starting from column <n> to the end of the record. If you do not specify this option, SORT begins sorting from column 1.

The most common use includes the command redirection operators < and > with file names (although any other redirection operator can be used). The input file is designated by < and the output file is designated as >.

Refer to Section 2 for more information about command input and output.

The format for this use of the SORT command is:

```
[d:][pathname]SORT [/R][/+n] [<[d1:]
[pathname1][filename1][.ext1]] [>[d2:]
[pathname2][filename2][.ext2]]
```

WHERE: [<[d1:][pathname1]

Specifies the drive and directory path of the file to be sorted.

filename1[.ext1]]

Specifies the file name and extension of the file to be sorted. The default is the console keyboard unless you precede SORT by another command and a piping symbol (|).

[>[d2:][pathname2]

Specifies the drive and directory path of the file to contain the sorted data.

filename2[.ext2]]

Specifies the file name and extension of the file to contain the sorted data. The default is the console display.

EXAMPLES: The following example reads the file UNSORT.TXT, reverses the sort, and then writes the output to a file named SORT.TXT:

```
SORT/R <UNSORT.TXT >SORT.TXT
```

The next command pipes the output of the directory command to the SORT filter. The SORT filter sorts the directory listing starting with column 14 (this is the column in the directory listing that contains the file size), then sends the output to the console. Thus, the result of this command is a directory sorted by file size:

```
DIR | SORT /+14
```

The command

```
DIR | SORT/+14 | MORE
```

does the same thing as the command in the previous example, except that the MORE filter gives you a chance to read the sorted directory one screen at a time. The maximum file size that can be sorted is 63K characters.

Although the SORT command can be a very useful tool, it has some limitations. Since it sorts according to the ASCII collating sequence, certain irregularities may appear within your sorted data.

- In ASCII, uppercase letters precede lowercase letters in the ascending order of a sort. In earlier versions of DOS, a lowercase a came after all uppercase letters, and uppercase Z preceded any lowercase letters. In this version of DOS, uppercase and lowercase letters are equivalent in value and are sorted together.

- Unless numbers have the same decimal placement and the same number of significant digits, a sorted list of numbers may prove inaccurate. Since sort judges all characters as letters, the number 1 precedes the number 2, but so do the numbers 10, 186, 1984, etc. However, if the numbers are 0001, 0002, 0010, 0186, and so on, the sort performs correctly.

- Special characters and control characters are sorted according to their ASCII value. Therefore, an end-of-file marker (Ctrl-Z) or a Tab (Ctrl-I) may destroy the integrity of your sort. Removing the special characters corrects that situation.

- You can sort keyboard input by entering the SORT command without any file names or pipe (|). The SORT command then waits for you to enter data from the keyboard. Press <ENTER> at the end of each line and F6 (Ctrl-Z) followed by <ENTER> when all input has been entered. The entered data is then sorted and displayed in the sorted sequence.

- Special international language characters (ASCII 128 to 175) are treated by the SORT command as if they are ASCII values for the U.S. character set as shown in the following table.

International Character			US Character Value for SORT		
ASCII Decimal	&H	Character	ASCII Decimal	&H	Character
128	80	Ç	67	43	C
129	81	ü	85	55	U
130	82	é	69	45	E
131	83	â	65	41	A
132	84	ä	65	41	A
133	85	à	65	41	A
134	86	å	65	41	A
135	87	ç	67	43	C
136	88	ê	69	45	E
137	89	ë	69	45	E
138	8A	è	69	45	E
139	8B	ï	73	49	I
140	8C	î	73	49	I
141	8D	ì	73	49	I
142	8E	Ä	65	41	A
143	8F	Å	65	41	A
144	90	É	69	45	E
145	91	æ	65	41	A
146	92	Æ	65	41	A
147	93	ô	79	4F	O
148	94	ö	79	4F	O
149	95	ò	79	4F	O
150	96	û	85	55	U
151	97	ù	85	55	U
152	98	ÿ	89	59	Y
153	99	Ö	79	4F	O
154	9A	Ü	85	55	U
155	9B	ø	36	24	$
156	9C	£	36	24	$
157	9D	¥	36	24	$
158	9E	Pt	36	24	$
159	9F	ƒ	36	24	$
160	A0	á	65	41	A
161	A1	í	73	49	I
162	A2	ó	79	4F	O
163	A3	ú	85	55	U
164	A4	ñ	78	4E	N
165	A5	Ñ	78	4E	N
166	A6	ª	166	A6	ª
167	A7	º	167	A7	º
168	A8	¿	63	3F	?
169	A9	⌐	169	A9	⌐
170	AA	¬	170	AA	¬
171	AB	1/2	171	AB	1/2
172	AC	1/4	172	AC	1/4
173	AD	i	33	21	!
174	AE	«	34	22	"
175	AF	»	34	22	"

88-747

STACKS

TYPE: Configuration.

PURPOSE: Allocates the number and size of system stacks used during interrupt processing.

FORMAT: STACKS=nn,sss

WHERE: nn

The number of stack frames to allocate (from 8 to 64, with the default value being 9).

sss

The size (in bytes) of each stack frame (from 32 to 512, with the default value being 128).

EXAMPLES: The following example allocates 10 stacks, each of which is 256 bytes (128 entries) long:

```
STACKS=10,256
```

The following example tells DOS not to install special stacks to handle interrupts:

```
STACKS=0,0
```

When set to zero, DOS does not intercept any interrupts, and uses only internal DOS stacks.

COMMENTS: When available stack resources are exceeded by a rapid succession of hardware interrupts, the fatal internal stack failure error message "Internal Stack Error" is displayed. DOS does not always have enough stack space available for interrupt routines. Network cards, in particular, can generate interrupts rapidly.

You can increase the stack size and number at system startup by adding a "STACKS=nn,sss" entry in the CONFIG.SYS file.

Increasing stack resources reduces available memory. You should increase the number of stacks first when trying to alleviate the stack error condition. Each time a hardware interrupt occurs, DOS allocates one frame from the stack pool. After the interrupt has been processed, DOS returns the frame to the stack pool.

SUBST

TYPE: External

PURPOSE: Substitutes a simple disk drive alias for a complex pathname.

FORMAT:
```
[d:][pathname]SUBST
    (to report the substitutes)
[d:][pathname]SUBST d1: d2:pathname2
    (to create the substitutes)
[d:][pathname]SUBST d1: /D
    (to delete the substitutes)
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the SUBST command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1:

Specifies the disk drive alias.

d2:pathname2

Specifies the drive and pathname to be substituted by alias.

/D

The request to delete the disk drive substitutes.

COMMENTS: The SUBST command allows you to specify a simple disk drive designator as an alias for a complex pathname. When DOS sees the drive alias, it substitutes the complex pathname the alias represents. This saves you the time and trouble of repeatedly entering the complex pathname.

The drive alias must not be the default drive, and must be permitted within the limits of the LASTDRIVE command parameter. The drive alias must therefore be A: through E: unless a larger LASTDRIVE parameter is specified in your CONFIG.SYS file. Also, all pathnames must originate in the root directory to be eligible for substitution.

For example, to substitute disk drive E: for the complex pathname C:\DOSLIB\VERSION3\UTILITIES. Enter the following command:

```
SUBST E: C:\DOSLIB\VERSION3\UTILITIES
```

You can now use E: instead of the more complex pathname. You should not use the following commands when a substitution is in effect: ASSIGN, BACKUP, DISKCOMP, DISKCOPY, FDISK, FORMAT, JOIN, LABEL, and RESTORE. You cannot substitute network drives or drives that were acted upon by the JOIN or ASSIGN commands. To avoid unexpected results, use all other DOS commands involving the drive alias with caution.

To display the substitutes in effect, enter the following:

```
SUBST
```

The following should appear:

```
E: => C:\DOSLIB\VERSION3\UTILITIES
```

To delete the substitution E: from the complex pathname, enter the following:

```
SUBST E: /D
```

SYS

TYPE: External

PURPOSE: Transfers the DOS hidden system files from the disk in the default drive to the disk in the drive specified.

FORMAT: [d:][pathname]SYS d1:

WHERE: [d:][pathname]

Indicates the drive and directory path where the SYS command is located. This option is not necessary if the command is in the current directory or you have previously defined a path to the command.

d1:

Specifies the drive to which the DOS hidden system files are to be transferred.

COMMENTS: SYS is normally used to update the DOS operating system files or to place the operating system files on a formatted 5-1/4 inch or 3-1/2 inch diskette or hard disk that contains no files. An entry for d1: is required. There must be enough space available at the beginning of the target disk to accommodate the system files, preferably through the use of the /S option with the FORMAT command, or there must be no files in the directory.

The transferred files are copied in the following order:

IBMBIO.COM
IBMDOS.COM

IBMBIO.COM and IBMDOS.COM are both hidden files that do not appear when the DIR command is executed. COMMAND.COM (the command processor) is not transferred. You must use the COPY command to transfer COMMAND.COM and any other external DOS commands that you may want on the new disk.

TIME

TYPE: Internal

PURPOSE: Displays or sets the time.

FORMAT: TIME [hh:mm[:ss[.cc]]] | [hh:mm[:ss[,cc]]]

WHERE:
hh = hour (00-24)
mm = minute (00-59)
ss = second (00-59)
cc = hundredths of a second (00-99)

COMMENTS: If you enter the TIME command without any parameters, the following message is displayed:

```
Current time is hh:mm:ss.cc
Enter new time:
```

Press the <ENTER> key if you do not want to change the time shown. A new time may be given as an option to the TIME command as in:

```
TIME 8:20
TIME 8.20
```

You must enter the new time using numerals only; letters are not allowed.

The hour, minute, and second entries must be separated by periods or colons. The hundredths of a second entry must be separated by a period only. You do not have to enter the mm (minutes), ss (seconds), or cc (hundredths of seconds) options.

DOS uses the time entered as the new time if the options and separators are valid. If the options or separators are not valid, DOS displays the message:

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

and waits for you to type a valid time.

TIME

You can change the format of the displayed time to an international format at system startup by using the COUNTRY command in the CONFIG.SYS file. You can also use the SELECT command to create a DOS system diskette that contains a CONFIG.SYS file with the desired COUNTRY command.

If you are using an AUTOEXEC.BAT file, you are not prompted to enter the correct time at the start of your session unless you include the TIME command in the AUTOEXEC.BAT file.

Leaving a PC on for longer than 24 hours without activity may cause the time to be updated incorrectly.

NOTE: In DOS revision 3.30, the DOS TIME command affects the setting stored by battery-operated clocks such as those in AP and SP Series PCs. To permanently set other battery-powered clocks, use the software provided with the clock option.

TREE

TYPE: External

PURPOSE: Displays all directory paths and subdirectory names on a disk and, if specified, all files contained within each subdirectory.

FORMAT: [d:][pathname]TREE [d1:][/F]

WHERE: [d:][pathname]

Indicates the drive and directory path where the TREE command is located. This option is not necessary if the command is in the current directory, or you have previously defined a path to the command.

d1:

Specifies the disk drive to be displayed.

/F

Requests the display of all the files in each directory.

COMMENTS: If you do not specify a disk drive, the default drive is used. The TREE command displays all subdirectories. With the /F option, it displays all file names in each subdirectory.

If there are many files and subdirectories on the disk, you may want to route the TREE display to a printer:

```
TREE > PRN
```

or to a file:

```
TREE > filename
```

You may also use the MORE filter to look at the tree one page at a time:

```
TREE | MORE
```

TYPE

TYPE: Internal

PURPOSE: Displays the contents of an ASCII file.

FORMAT: TYPE [d:][pathname]filename[.ext]

WHERE: [d:][pathname]

Specifies the drive and directory path of the file to be displayed.

filename[.ext]

Specifies the file name and extension of of the file to be displayed.

COMMENTS: Use this command to examine an ASCII file without modifying it. (Use DIR to find the name of a file, and EDLIN to alter the contents of a file.) You cannot display a directory, nor can you use the wildcard characters * and ?. You can use the MORE filter with this command to view the contents of a file a page at a time:

```
TYPE filename.ext | MORE
```

The only formatting performed by TYPE is that tabs are expanded to spaces consistent with tab stops every eighth column. You should not try to display binary files, because some non-ASCII control characters (such as Ctrl-Z) are sent to your PC, causing bells, form feeds, and escape sequences to be executed. The display can be redirected to another output device such as a printer (>PRN), or you can print it after it appears on the screen by pressing <Ctrl-PrtSc> before issuing the TYPE command.

VDISK.SYS

TYPE: Configuration

Refer to the DEVICE command description.

VER

VER

TYPE: Internal

PURPOSE: Displays the current DOS version number.

FORMAT: VER

COMMENTS: This command displays the current version of the DOS system that is loaded in your PC. Only the first three digits of the version number are displayed. For example:

```
MS-DOS Version 3.30
```

VERIFY

TYPE: Internal

PURPOSE: Turns the verify switch on or off (when writing to disk).

FORMAT: VERIFY [ON | OFF]

WHERE: ON

Sets verification on.

OFF

Turns verification off.

COMMENTS: If you want to be assured that all files are written correctly to disk, you can use the VERIFY ON command to tell DOS to read back each file that it writes to disk (to check for bad sectors, for example). DOS then performs a VERIFY each time data is written to a disk. You receive an error message only if DOS was unable to successfully reread the data written to the disk. Because of the extra time it takes to verify, DOS operates more slowly with VERIFY ON.

VERIFY ON remains in effect until you issue a VERIFY OFF command.

To display the current setting of VERIFY, type VERIFY with no options. The default for VERIFY is OFF.

The VERIFY command has the same purpose as the /V option in the COPY command.

This command is not valid while writing to a network disk.

VOL

TYPE: Internal

PURPOSE: Displays the disk volume identifier, if it exists.

FORMAT: VOL [d:]

WHERE: d:

Specifies the disk drive containing the volume label you want to read.

COMMENTS: This command displays the volume ID (label name) of the disk in the drive you specify. If you do not specify a drive, DOS displays the volume ID of the disk in the default drive.

XCOPY

TYPE: External

PURPOSE: Selectively copies files and directories, including lower level directories if they exist.

FORMAT:
```
[d:][pathname]XCOPY
[d1:][pathname1][filename1[.ext1]]
[d2:][pathname2][filename2[.ext2]]
[/A][/D:date][/E][/M][/P][/S][/V][/W]
```

WHERE: [d:][pathname]

Indicates the drive and directory path where the XCOPY command is located. This option is not necessary if the command is in the current directory or you have previously defined a path to the command.

[d1:][pathname1]

Specifies the drive and directory path of the file(s) or directory to be copied.

[filename1[.ext1]]

Specifies the file name and the extension of any source file(s) to be copied. The wildcard characters (* and ?) are permitted.

[d2:][pathname2]

Specifies the drive and directory path of the target file(s) or directory.

[filename2[.ext2]]

Specifies the file name and the extension of any target file(s).

/A

Copies source files that have their archive bit set. Does not modify the archive bit of the source file. (Refer to the ATTRIB command description for information on setting the archive attribute.)

/D:

Copies source files that you modified on or after the date specified by <mm-dd-yy> (or as defined for international date/time formats).

/E

Used with the /S option to copy any subdirectories, even if they are empty.

/M

Copies source files that have their archive bit set (such as the /A option), but turns off the archive bit in the source file.

/P

Prompts you with (Y/N), allowing you to confirm whether you want to create each target file.

/S

Copies directories and lower level subdirectories, unless they are empty (use /E for empty subdirectories). If you omit this option, XCOPY works within a single directory. The new directory created does not retain the date of the source directory.

/V

Causes XCOPY to verify each file as it is written to the target, to make sure that the target files are identical to the source files.

/W

Causes XCOPY to wait for you to insert a diskette before it starts copying files. You must press a key to continue or press <Ctrl-C> to abort the XCOPY command.

COMMENTS: Because the DISKCOPY command copies disks track by track, it requires your source and target disks to have the same format. If you want to copy a disk that contains files in subdirectories to a target disk that has a different format, you must use the XCOPY command. For example:

```
XCOPY  A:  B:  /S  /E
```

copies all of the files and subdirectories (including any empty subdirectories) on the disk in drive A to the disk in drive B.

EXAMPLES: To copy all of the directories, files, and subdirectories (unless they are empty) on drive C to drive D, type:

```
XCOPY  C:\  D:\  /S
```

To copy all of the files and subdirectories within SUB1 to the root directory in drive D, type:

```
XCOPY  C:\SUB1  D:\  /S
```

To copy all of the subdirectories within SUB1 to the directory SUB2 on drive D, type:

```
XCOPY  C:\SUB1  D:\SUB2  /S
```

The following information should be noted about the XCOPY command:

- If you do not specify a pathname, XCOPY begins from the current directory.
- The default file name is *.*.
- If the directory path that you specified does not exist on the target disk, it is created before the files are copied.
- You can rename files (as in the COPY command) by specifying new names for the target files.
- Unlike the COPY command, XCOPY cannot copy to or from the reserved device names CON and LPT1.

- The total number of characters in a pathname (including the drive designator, pathname, file name, and file extension) cannot exceed 63 characters at any time while searching or copying.

- The XCOPY command cannot copy hidden or deny-read files from the source, or copy read-only or deny-write files on the target.

- If the source and target pathnames are the same, the following messages may be displayed:

  Access denied

  or:

  File cannot be copied onto itself

- If the target directory is part of the source pathname, the following message is displayed:

  Cannot perform cyclic copy

The /A option copies those files that have been changed since the last XCOPY /M or BACKUP /M command, but unlike the /M option, does not change the setting of the archive attribute.

The /D option uses the date format set for the appropriate country code by the COUNTRY or SELECT commands. The U.S. format is MM-DD-YY.

The /E option permits the copying of empty subdirectories when you are re-creating directory trees with the /S option.

The /M option allows you to use XCOPY in backing up files. If the archive attribute of the file is set, it means that the file has been created or modified since the last XCOPY /M or BACKUP /M command. You can also set the archive attribute with the ATTRIB command.

The /P option allows copying on a file-by-file basis. Each file name is displayed, so that you can select whether or not to copy it:

d:pathname\filename.ext (Y/N)?

The /S option begins at the current directory and copies all lower subdirectories and files. If you used the /E option, empty subdirectories are copied. If you did not use the /E option, empty subdirectories are omitted.

The /V option verifies the copy as it is made. This option causes the XCOPY command to run more slowly.

The /W option displays the following message before beginning the copy:

Press any key to begin copying file(s)

To copy several directories or an entire disk to diskettes that do not have enough space, use the /S and /M options. When the target diskette is full, the system displays "Disk full," and stops. Insert a second diskette and issue the same XCOPY command. Because the /M option turns off the archive bit, only those files that have not yet been copied are copied. Repeat these steps until all files are copied. (Remember that the archive bit has been turned off for all of the copied files.)

NOTE: Unlike the BACKUP command, the XCOPY command cannot copy a file that is bigger than the available size on the target diskette.

Section 4
EDITING, FUNCTION KEYS, CURSOR, DISPLAY GRAPHICS

In this section:

SPECIAL EDITING KEYS

Special editing keys allow you to repeat all or part of the last line you typed on the screen. Each line you enter is copied by DOS into a special one-line storage area. When you use the special editing keys to re-create the last line you typed, the characters are copied from the storage area into the current line. You can selectively copy all or part of the last stored line into the current line, and insert additional characters where needed.

You can use special editing keys in two areas:

- You can use special editing keys when you enter DOS commands, as described in this section.
- You can use special editing keys to edit lines of data within source programs or text files, as defined in Section 5.

The special editing keys are:

- The Delete <Del> key
- The Escape <Esc> key
- The Insert <Ins> key
- The first five programmable function keys at the left side of the keyboard (<F1> through <F5>).

Special editing keys and their functions are described in Table 4-1.

Programming The Editing Keys

DOS automatically programs the meanings shown in Figure 4-1 to function keys <F1> through >F5>. Function key <F6> is also programmed to signal the end of certain keyboard entry operations. The use of this key is described in the *MS-DOS User's Guide* (Order No. HU94), and in the description of the COPY command in Section 3.

You can, however, program function keys <F1> through <F6> to perform other functions. You can also program other function keys to perform functions normally assigned to keys <F1> through <F6>. This procedure is described later in this section.

IMPORTANT

If you program an editing function to a different key or reprogram keys <F1> through <F6>, the descriptions of the use of that key in this section and in Section 5 are no longer accurate.

Table 4-1. Special Editing Functions

Function	Key	Description
Copy one character	<F1> or →	Copies one character from the stored line to the command line.
Copy up to character	<F2>x	Copies characters up to the character (x) specified from the stored line to the command line.
Copy stored line	<F3>	Copies all remaining characters from the stored line to the command line. (Copies the last command entered.)
Skip one character	<Del>	Skips over (does not copy) one character in the stored line.
Skip up to character x	<F4>x	Skips over (does not copy) the characters in the stored line up to the character (x) specified.
Quit input	<Esc>	Voids current input line. Leaves stored line unchanged.
Insert and Replace mode	<Ins>	Enters insert mode. Press again to enter replace mode.
Store new line	<F5>	Makes the new line the stored line.
End of Keyboard entry	<F6>	Signals the end of command line for certain keyboard entry operations.

Editing DOS Command Lines

You can use the special editing keys to edit a command line in the following ways:

- You can repeat the last command line you entered by pressing the <F3> key (copy remainder of stored line) and the <ENTER> key.
- If you make a mistake in the command line, you can edit it and retry without having to retype the entire command line.
- You can edit and execute a command line that is similar to the last command line you typed, with a minimum amount of typing.

The relationship between the command line and the line storage area is shown in Figure 4-1.

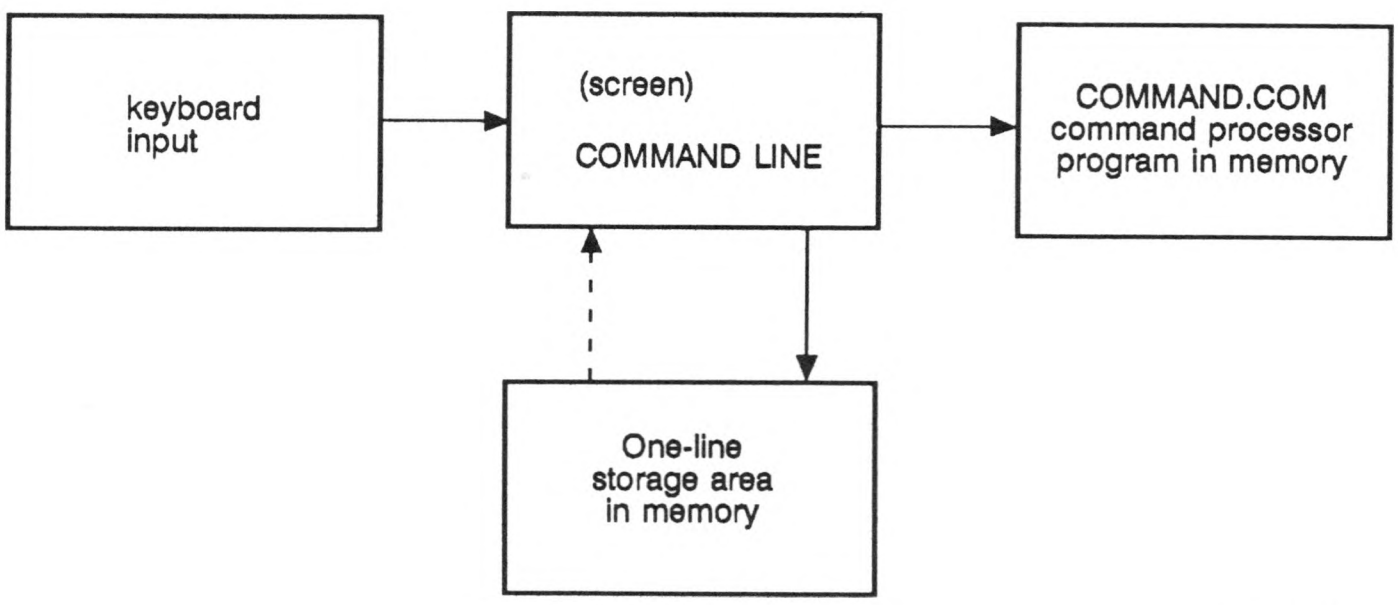


Figure 4-1. Command Line Usage

When you type a DOS command at the DOS prompt, and press the <ENTER> key, the command is automatically sent to the command processor (COMMAND.COM) for execution. At the same time, a copy of the command is stored in the special one-line storage area (replacing the previous stored line). You can now recall the command or modify it with the DOS special editing keys.

The following examples describe the use of the editing keys. Each of these keys is more fully described in Section 5, where they are described for use in editing source programs or text files.

Examples

If you type the command:

```
DIR PROG.COM
```

DOS displays information from the current directory about the file PROG.COM. The command line is also copied into the special storage area. You can now edit this stored line in different ways:

1. To repeat the command with no change, type:

   <F3><ENTER>

   The command is redisplayed at the DOS prompt:

   DIR PROG.COM

   Function key <F3> copies all remaining characters from the stored line to the current command line. Since this is the first editing key you have used, the copy begins with the first character of the stored line. A complete duplicate of the last command line you typed is displayed. Press <ENTER> to execute the new command line, and copy it into the special storage area.

2. To display directory information about a file named PROG1.COM, first type:

   <F2>.

   DOS copies all characters from the stored line up to, but not including, the period. The command line now reads:

   DIR PROG

   Now type:

   <Ins>1

   Insert mode is entered, and a 1 is inserted. The command line now reads:

   DIR PROG1

   Now type:

   <F3><ENTER>

Press any function key to turn off insert mode (the <Del> key does not). The <F3> key copies the remainder of the stored line. The command line now reads:

DIR PROG1.COM

Press <ENTER> to execute the command line, and copy it into the special storage area.

Insert mode was required to permit insertion of the 1 before the period. If you had not used the <Ins> key, the default condition, replacement mode, would have been in effect. In replacement mode, the 1 entry would have been taken as a replacement for the period; the subsequent <F3> copy would have begun with the character following the period, and the resulting command line would have read:

DIR PROG1COM.

3. To display directory information about a file named PROG.ASM, first type:

   <F2>1

   DOS copies all characters from the stored line up to, but not including, the 1 inserted during the previous example. The display now reads:

   DIR PROG

   Now type:

   <Del>

   This skips over the "1" in the stored line. The displayed command line does not change.

   Next type:

   <F1>

   This causes the copy of one character, the period, from the stored line. The display now reads:

   DIR PROG.

The same result can be obtained by typing:

<F2>C

This entry (copy up to the C) can be used instead of <F1> because in this particular instance only the period is copied.

Now type:

ASM

The result is:

DIR PROG.ASM

Pressing <ENTER> causes this command line to be sent to the command processor for execution and to be copied into the special storage area.

The COM extension previously in the stored line is not copied, skipped, or affected by any of the editing functions. It is then destroyed when the revised command is copied into the storage area, replacing the old contents.

You can use another alternate sequence of entries to accomplish the same results as this example. In the alternate sequence, which is shown last because it illustrates the use of fewer special editing keys, type:

<F2>1

as before. After the display:

DIR PROG

type:

.ASM

The display is now:

DIR PROG.ASM

To execute and copy this command, press <ENTER>.

4. Now assume that you want to display the contents of the file whose directory information was displayed in the previous step. You want to modify the previous command to read:

   TYPE PROG.ASM

   To do this, type:

   TYPE<Ins><space><F3><ENTER>

   (where <space> represents entry of a single space character)

   Notice that when you type the letters TYPE, these characters go directly into the command line. Since you are in replacement mode (rather than insert mode), these characters effectively overwrite or replace the corresponding characters (DIR<space>) in the stored line. The next editing function, which references the stored line, now begins with the P.

   The <Ins> and the <space> character are needed now to insert a space after TYPE before copying PROG.ASM into the command line. <F3> copies the remainder of the stored line (PROG.ASM) into the command line. <ENTER> causes the command line to be executed and to be copied into the special storage area. (<F3> turns off insert mode and returns DOS to replace mode.)

5. If at this point you decide that you want to see directory information for all assembler language source programs in the subdirectory PAYROLL of the subdirectory PROGSRCE on this disk, type:

   <F4><space><Del><Ins>DIR<space>
   \PROGSOURCE\PAYROLL\<Ins>
   <F2>.<Ins>*<F3>

   <F4> skips over the characters in the stored line up to the space. The other editing keys function as previously described. The command line now displays:

   DIR \PROGSOURCE\PAYROLL\PROG*.ASM

6. You realize that you forgot to abbreviate SOURCE in the first subdirectory name. To fix this, first type:

   <F5>

   This copies the new command line into the special storage area without executing it. DOS identifies the results of this function by leaving the command line (the new stored line) displayed on the screen followed by an @. The cursor is then positioned to start a second line for the new command line. The display now appears:

   DIR \PROGSOURCE\PAYROLL\PROG*.ASM@

   Now type:

   <F2>S<F1><Del><Del><F3><ENTER>

   <F2>S copies up to the S. (Your first inclination would be to copy up to the O in SOURCE. However, a single <F2> would only copy up to the O in PROG, which is not what you want.) <F1> copies the S (note that the sequence <F2>O<F2>O would be equivalent to <F2>S<F1>). The <Del><Del> sequence skips the OU characters. <F3> copies the remainder of the latest stored line into the command line. The display is now:

   DIR \PROGSOURCE\PAYROLL PROG*.ASM@
   DIR \PROGSRCE\PAYROLL\PROG*.ASM

   Pressing <ENTER> causes the new command (bottom) line to be executed and to be copied into the special storage area.

7. Next you may decide to display the contents of one of the payroll source programs found in the directory display during the previous example.

   To do this, you type:

   <Ins>TYPE<F4><space><F2>*<Del><Ins>W2<F3>

   The display now reads:

   TYPE \PROGSRCE\PAYROLL\PROGW2.ASM

8. If, before you press <ENTER>, you change your mind and decide you would rather see the directory display of your general ledger source programs first, type:

   <Esc>

   <Esc> voids the current input line and leaves the special one-line storage area unchanged. DOS identifies the result of this action by showing a backslash (\) following the line you voided. The display now reads:

   TYPE \PROGSRCE\PAYROLL\PROGW2.ASM

   Since the stored line still contains the last DIR ... entry, you now type:

   <F2>P<F2>P<Del>LEDGER<F3>

   The two <F2>P functions copy up to the word PAYROLL in the DIR ... command; the <Del> skips the letter P; the letters LEDGER replace the letters AYROLL; and the <F3> function copies the remainder (\PROG*.ASM) of the stored line. The display now reads:

   TYPE \PROGSRCE\PAYROLL\PROGW2.ASM
   DIR \PROGSRCE\LEDGER\PROG*.ASM

   Pressing <ENTER> now causes the new command (bottom) line to be executed and to be copied into the special storage area.

PROGRAMMING THE KEYBOARD

This section tells you how to program individual keys of your keyboard. You will probably want to use this facility for programming the function keys or the Alt mode of particular keys.

First, you have to install the ANSI driver (provided on your MS-DOS diskettes) in the CONFIG.SYS. The file name of this driver is ANSI.SYS.

If CONFIG.SYS does not yet exist, create a new file with that name (refer to Section 8). Using EDLIN or another editor, add the following line to the CONFIG.SYS file:

```
DEVICE=[d:][pathname]ANSI.SYS
["comment"][buffer length]
```

Where:

[d:][pathname]

Defines the drive and directory path where the ANSI.SYS file can be found. This option is not necessary if the file is in the root directory.

"comment"

A string enclosed in quotation marks (such as, "buffer length =").

buffer length

Length of the buffer used by DOS to store the new function key definitions that you define with the PROMPT (or ESC) command. This value can be in the range 200 to 9999 bytes, with the default 200 bytes.

If you are using a hard disk, and you placed the ANSI.SYS file in the \SYS directory, then you must specify the directory path as well:

```
DEVICE=C:\SYS\ANSI.SYS
```

In this example, \SYS indicates that the ANSI.SYS file is listed in a directory named SYS which is a subdirectory of the root directory. Typically, diskette users place the device drivers in the root directory.

Programming the Function Keys

After you have installed the ANSI.SYS file in CONFIG.SYS, you can program function keys using the PROMPT command. You normally use the PROMPT command to alter the system prompt, but in this instance you can also use it to alter the character belonging to a particular key. The general format of this command is:

```
PROMPT $e[0;#;"string";0;#...p
```

You must type the $, lowercase e, and [characters as true characters. The # represents a number (the code for the key you wish to program). Following a semicolon with the content you wish to assign to a particular key. This may be a string enclosed in quotation marks (for example "DIR"), or a direct (ASCII) value (such as 13, for <ENTER>), or any combination of string and direct value items. Separate the items from each other by semicolons. You do not end the last item with a semicolon, however, but by a lowercase p.

Normally, you use this form of the PROMPT command in order to program a function key. In this case, you should bear in mind that the code for a function key consists of not one but two numbers. The first of these two numbers is always zero. The second number is the one that selects a particular Function Key, including shifted (uppercase), Ctrl and Alt modes:

0; 59...68	Function Keys <F1>...<F10> in base mode
0; 84...93	Function Keys <F1>...<F10> in shifted (uppercase) mode
0; 94...103	Function Keys <F1>...<F10> in Ctrl mode
0;104...113	Function Keys <F1>...<F10> in Alt mode

Programming Alternate Mode of Alphabetic Keys

If you wish to program an alphabetical key in the Alt mode, you have to select an appropriate second number from the following:

0;16...25	Alt Q, W, E, R, T, Y, U, I, O, P
0;30...38	Alt A, S, D, F, G, H, J, K, L
0;44...50	Alt Z, X, C, V, B, N, M

Restoring the System Prompt

After you have defined one or more keys using the PROMPT command, the system prompt disappears from the screen. To restore it, issue a second prompt command, this time without any characters or specific characters you want to see in the prompt, or define the prompt at the end of the PROMPT command line that defines the key ($...).

The new key assignment remains until you reload the operating system. If you frequently need a specific set of key assignments, it is a good idea to store a set of corresponding PROMPT commands in a batch file for execution immediately after loading the operating system.

Examples

The following PROMPT command assigns the DOS command "DIR" with a subsequent <ENTER>, to Function Key <F10>:

```
PROMPT $e[0;68;"DIR";13p
```

To give the <Alt A> key combination the DOS command "TYPE" followed by <ENTER>, issue the following command:

```
PROMPT $e[0;30;"TYPE";13p
```

PROGRAMMING THE CURSOR AND DISPLAY GRAPHICS

You can change the location and appearance of the cursor and the appearance of the display screen by using the PROMPT $e[... format of the PROMPT command. The $e used with the PROMPT command defines the ESC (hexadecimal 1B) sequence. Include the DEVICE=ANSI.SYS command in the CONFIG.SYS file for the cursor and display change functions to work. The ANSI.SYS acts as an extended keyboard and screen device driver in this case, and supports the use of the $e (ESC) sequence with the PROMPT command.

Exercise extreme caution when using these functions. Your errors can cause unpredictable results and bring down the system. The following discussion uses the PROMPT command.

Several cursor and display functions can be combined on a single line to perform a sequential series of functions using only one PROMPT $e[... command. This series of functions must end with a function to set the prompt, or in some cases, the very next command must be a PROMPT command to set the prompt. If you do not set the prompt, no prompt is displayed after the function is executed. Do not combine functions to set the function keys with functions of any other type function within one PROMPT $e[... command. Several functions can be combined to set multiple Function keys on one line, or several cursor and display functions can be combined on one line; but do not combine the two types of functions.

Use exactly the same case of alphabetic character, as is defined for each function. If an uppercase or lowercase character is defined, do not use one character for another; such as, h for H.

Default values are defined for each variable associated with a function. The default values for a function variable are used if no other value is entered or if you enter a value of zero for the variable.

The PROMPT $e[... command can be entered from the DOS command line. The more typical usage for this command is to have several PROMPT commands that set function keys, place the cursor, set color attributes, and define the prompt, in the AUTOEXEC.BAT file. Entering an individual PROMPT command for a cursor movement function from the command line, has little practical use.

The cursor and display graphics functions can be used with the PROMPT command, or these functions can also be used as control sequences when issued by DOS programming function calls that can write to an output device (the keyboard and display screen).

Programming the Cursor

The cursor functions operate as indicated when you use them as control sequences for DOS programming function calls that write to a standard output device. The format is:

```
PROMPT $e[...
```

When you use these functions with the PROMPT command, you should note the following:

- For cursor movement or status functions, the function is performed after the PROMPT command is entered. Therefore, the position of the cursor is at the beginning of the next line when the cursor movement or status function is executed.

- If a cursor movement function is executed and not immediately followed by a function to define the prompt on the same line, then each time you press the <ENTER> key, the PROMPT command you entered is executed again. This condition continues in effect only if or until the entire screen pages to a clear screen, then the PROMPT command is no longer in effect. If you do not define the prompt, no prompt text is displayed after the PROMPT function is executed.

- You should follow the PROMPT function with a function to define the prompt on the same line. This causes both the cursor movement or status function and prompt definition function to remain in effect until another PROMPT command is entered. After you press the <ENTER> key, DOS executes the command that you entered on the command line first, then the PROMPT $e[... command that you entered.

Command	Function	Description
PROMPT $e[#;#H	Cursor Position	Moves the cursor to the position specified by the two parameters. The first parameter specifies the line number and the second parameter specifies the column number. The default value is 1, which moves the cursor to the home position.
PROMPT $e[#A	Cursor Up	Moves the cursor up the number of lines specified. The default value is 1. This function is ignored if the cursor is already on the top line.
PROMPT $e[#B	Cursor Down	Moves the cursor down the number of lines specified. The default value is 1. This function is ignored if the cursor is already on the bottom line.
PROMPT $e[#C	Cursor Forward	Moves the cursor forward the number of columns specified. The default value is 1. This function is ignored if the cursor is already in the rightmost column.
PROMPT $e[#D	Cursor Backward	Moves the cursor back the number of columns specified. The default value is 1. This function is ignored if the cursor is already in the leftmost column.
PROMPT $e[#;#f	Horizontal and Vertical Position	Moves the cursor to the position specified by the two parameters. The first parameter specifies the line number and the second parameter specifies the column number. The default value is 1, which moves the cursor to home position (same as the Cursor Position function).

Command	Function	Description
PROMPT $e[s	Save Cursor Position	Saves the current cursor position. This cursor position can be restored with the Restore Cursor Position function.
PROMPT $e[n	Restore Cursor Position	Restores the cursor to the value it had when the console driver received the Save Cursor Position function.
PROMPT $e[2J	Erase Display	Erases all of the screen and the cursor goes to the home position.
PROMPT $e[k	Erase Line	Erases from the cursor to the end of the line, including the cursor position.

Programming Display Graphics

The display graphics functions operate as indicated when used as control sequences for DOS programming function calls that write to a standard output device. The format is:

```
PROMPT $e[...
```

You should follow the PROMPT function with a function to define the prompt on the same line. This causes both the display graphics function and prompt definition function to remain in effect until you enter another PROMPT command. After you press the <ENTER>, DOS executes the command you entered on the command line first, then the PROMPT $e[... command that you entered. If you do not define the prompt, no prompt text is displayed after the PROMPT function is executed.

Command	Function	Description
PROMPT $e[#;...;#m	Set Graphic Rendition	Sets the attribute specified by the # parameter(s). All screen and display characters have the attributes defined by the # parameter(s), until this function is entered again. Parameters 30 through 47 apply to color graphic displays only.

Parameter Number	Description
0	All attributes Off (default: white on black)
1	Bold On (high intensity)
4	Underscore On (Monochrome Display only)
5	Blink On
7	Reverse video On (Black and White)
8	Cancelled On (invisible)
30	Black foreground
31	Red foreground
32	Green foreground
33	Yellow foreground
34	Blue foreground
35	Magenta foreground
36	Cyan foreground
37	White foreground
40	Black background
41	Red background
42	Green background
43	Yellow background
44	Blue background
45	Magenta background
46	Cyan background
47	White background

Command	Function	Description
PROMPT $e[=#h	Set Mode	Defines the screen width or type specified by the # parameter. You must follow this function with a second PROMPT command to set the prompt, or the prompt returns to the same starting position each time you press the <ENTER> key. Functions 0 through 6 apply to color graphic displays only.

Parameter Number	Description
0	40 columns x 25 lines black and white
1	40 columns x 25 lines color
2	80 columns x 25 lines black and white
3	80 columns x 25 lines color
4	320 x 200 pixels color (no cursor)
5	320 x 200 pixels black and white (no cursor)
6	640 x 200 pixels black and white (no cursor)
7	Sets wrap at end of line (typing past end-of-line wraps around to the beginning of the next line. You can enter characters until the keyboard input buffer is full). This is the DOS default mode.

Command	Function	Description
PROMPT #e[=#1	Remove Mode	Removes the screen width of type specified by the # parameter. Parameters are the same as the Set Mode function except that parameter 7 turns off wrap at end-of line. (You cannot enter characters past the end of line.) You must follow this function with a second PROMPT command to set the prompt, or the prompt returns to the same starting position each time you press the <ENTER> key.

Examples

The following example displays the date and time in the upper-right corner of the screen in reverse video and sets the prompt to the current pathname preceded by < and followed by >. This series of functions is executed by DOS each time you press the <ENTER> key (following the execution of the command entered on the command line).

```
PROMPT $e[s$e[1;50f$e[7m$d/$t$e[0m$e[u$1$p$g
```

- The $e[s function saves the current position of the cursor.
- The $e[1;50f positions the cursor on line 1 column 50.
- The $e[7m sets the reverse video.
- The $d/$t gets the date and time.
- The $e[0m sets all attributes off (to turn off the reverse video).
- The $e[u restores the cursor to the position saved earlier.
- The 1p$g sets the cursor to "<pathname>".

If you have a color graphic display, you could modify the above PROMPT command to set your display color attributes also. The following example sets the background to blue (44) and the foreground to yellow (33) (after resetting all attributes ($e[0m) following the reverse video ($e[7m) of the time/date display).

```
PROMPT
$e[s$e[1;50f$e[7m$d/$t$e[0;33;44m$e[u$l$p$g
```

These PROMPT commands, or others of your own design, can easily be included in an AUTOEXEC.BAT file and executed each time the system is started.

Section 5
LINE EDITOR (EDLIN)

In this section:

Summary

This section provides instructions for using the MS-DOS Line Editor (EDLIN) to create, change, and display files.

GENERAL INFORMATION

You can use EDLIN to create, change, and display files, whether they are source program or text files. Specifically, you can use EDLIN to perform the following:

- Create new source files and save them.
- Update existing files and save both the updated and original files.
- Delete, edit, insert, and display lines.
- Search for, delete, or replace text within one or more lines.

The text in files created or edited by EDLIN is divided into lines, each up to 253 characters long. Line numbers are generated and displayed by EDLIN during the editing process, but are not actually present in the saved file.

When you insert lines, all numbers following the inserted text advance automatically by the number of lines you insert. When you delete lines in a file, all line numbers following the deleted text decrease automatically by the number of lines deleted. As a result, lines are always numbered consecutively in your file.

How to Start EDLIN

To start EDLIN, type:

```
[d:][pathname]EDLIN
[d1:][pathname1]filename1[.ext1][/B]
```

Where:

[d:][pathname]

Indicates the drive and directory path where EDLIN can be found. This option is not necessary if EDLIN is in the current directory, or you have previously defined a path to the command.

[d1:][pathname1]

Indicates the drive and directory path of the file to be edited.

filename1[.ext1]

Indicates the file name and extension of the file to be edited.

/B

Requests that the embedded <Ctrl-Z> characters (end-of-file markers) be ignored.

If you are creating a new file, the filename1[.ext1] should be the name of the file you want to create. If EDLIN does not find this file on a drive, it creates a new file with the name you specify. The following message and prompt are displayed:

```
New file
*
```

Special Editing Keys

The special editing keys that are described in Section 4 are also useful when you are in EDLIN. Table 5-1 contains a description of these keys and their functions:

Table 5-1. Special Editing Keys

Key	Function Performed
Del	Skips over (does not copy) one character in the stored line.
Esc	Voids the current input line; leaves the stored line unchanged.
Ins	Enters insert mode. Press <Ins> again to exit mode when done. (Second <Ins> is not needed if <F1>-<F5> pressed.)
F1 or →	Copies one character from the stored line to the new changed line, each time it is pressed.
F2x	Copies the stored line up to the character (x).
F3	Copies all remaining characters from the stored line to the new line.
F4x	Skips over (does not copy) characters up to the (x).
F5	Makes the new line the stored line: moves the cursor to the beginning of the next line, ready for more changes.

The prompt for EDLIN is an asterisk (*). When the prompt is displayed, you can type lines of text into your new file. To begin entering text, you must enter an I (Insert) command to insert lines. The I command is discussed later in this section.

To edit an existing file, the filename[.ext] you enter should be the name of the file you want to edit. When EDLIN finds the file you specify on the designated or default drive, the file is loaded into memory. If the entire file can be loaded, EDLIN displays the following message on your screen:

```
End of input file
*
```

You can then edit the file using EDLIN editing commands.

If the file is too large to be loaded into memory, EDLIN loads lines until memory is three-fourths full, and then displays the * prompt. You can then edit the portion of the file that is in memory.

To edit the remainder of the file, you must save some of the edited lines on disk in order to free memory. EDLIN then loads the unedited lines from disk into memory. Refer to the Write and Append command descriptions in this section for the procedure.

When you complete the editing session, you can save the original and the updated (new) files by using the End command. The End command is discussed in this section under "EDLIN Commands." The original file is renamed with an extension of .BAK, and the new file has the file name and extension you specified in the EDLIN command. The original .BAK file is not erased until you exit EDLIN, or until disk space is needed by EDLIN.

NOTE: Do not try to edit a file with a file name extension of .BAK, because EDLIN assumes that any .BAK file is a backup file. If you find it necessary to edit such a file, rename the file with another extension (using the DOS RENAME command discussed in Section 3), then start EDLIN and specify the new filename.ext.

EDLIN COMMANDS

Following are descriptions of the EDLIN commands that perform editing functions on lines of text. You can use these commands at the * prompt, but not while in Edit mode. (You enter Edit mode when you use the Edit line-number command. Refer to "line-number" in this section.) Before using an EDLIN command, read the conventions and options that apply to all commands.

Format Conventions

- Pathnames are acceptable as options to commands. For example, type EDLIN \USERS\SUE\TEXT.TXT to edit the TEXT.TXT file in the subdirectory SUE.

- You can reference line numbers relative to the current line (the line with the asterisk).

  - Use a minus sign with a number to indicate lines before the current line.

  - Use a plus sign with a number to indicate lines after the current line.

  Example:

  ```
  *-10,+10L
  ```

  This command line lists 10 lines before the current line, the current line, and 10 lines after the current line.

- You can enter multiple commands on one command line. When you issue a command to edit a single line using a line number (n), you must use a semicolon to separate commands on the line. Otherwise, one command may follow another without any special separators.

  Example:

  ```
  *15;-5,+5L
  ```

NOTE: In the case of a Search or Replace command, the string may be ended by an <F6> or <Ctrl-Z> instead of <ENTER>.

The command line in the next example searches for "This string" and then displays five lines before and five lines after the line containing the matched string. If the search fails, the displayed lines are those line numbers relative to the current line.

```
*S This string <F6>-5,+5L
```

- You can type EDLIN commands with or without a space between the line number and command. For example, to delete line 6, the command 6D is the same as 6 D.

- You can insert a control character (such as <Ctrl-Z>) into text by using the quote character <Ctrl-V> before it while in insert mode. <Ctrl-V> tells DOS to recognize the next uppercase letter typed as a control character. You can also use a control character in any of the string arguments of Search or Replace by using the special quote character. For example:

  *S<Ctrl-V>Z

  finds the first occurrence of <Ctrl-Z> in a file.

  *R<Ctrl-V>Z<Ctrl-Z> AND

  replaces all occurrences of <Ctrl-Z> in a file with AND.

  To insert <Ctrl-V> into the text, type <Ctrl-V>V.

- The <Ctrl-Z> (<F6> key) character ordinarily means, "This is the end of the file." If you have <Ctrl- Z> characters elsewhere in your file, you must tell EDLIN that these other control characters do not mean "End of File." Use the /B option when starting to tell EDLIN to show you the entire file including any embedded <Ctrl-Z> characters.

- <line> when used in this text refers to the line number. Do not type "line," just the number of the desired line.

Command Options

Several EDLIN commands accept one or more options. The effect of a command option varies, depending on the command. The following list describes each option:

line

<line> indicates a line number that you type. Line numbers must be separated by a comma or a space from other line numbers.

<line> may be specified in one of three ways:

Number (n) -- Any number less than 65529. If you specify a number larger than the largest existing line number, <line> means the line after the last line number.

Period (.) -- If you specify a period for <line>, it means the current line which is marked on your screen by an asterisk (*), between the line number and the first character.

Pound (#) -- The pound sign indicates the line after the last line number. If you specify # for <line>, it has the same effect as specifying a number beyond the last line number.

<ENTER>

Pressing the <ENTER> key without any of the <line> specifiers, directs EDLIN to use a default value appropriate to the command.

?

You use the question mark only with the Replace and Search commands. This option directs EDLIN to ask you if the correct string has been found. Before continuing, EDLIN waits for you to press either a Y or <ENTER> for a yes response, or any other key for a no response.

string

You use the <string> option only with the Search and Replace commands. A <string> (in the Search command) or <string1> (in the Replace command) represents text to be found, and <string2> (in the Replace command) is the text to replace other text. Each <string> must end with an <F6> or <ENTER>. (Refer to the Replace command for details.) Do not leave spaces between strings, or between a string and its command letter, unless you want those spaces to be part of the string.

n

You can use the n option to specify the number of lines to read from disk or write to disk with the Append (read) or Write command.

COMMAND DESCRIPTIONS

Following are detailed descriptions of each of the EDLIN commands. Refer to Table A-4 in Appendix A for a summary of EDLIN commands.

LINE-NUMBER

PURPOSE: Enters Edit mode, and specifies line of text to be edited.

FORMAT: [line]

COMMENTS: When you type a line number, EDLIN displays the line number and text, then reprints the line number on the line below. The line is now ready for editing. You can use any of the EDLIN special editing keys to edit the line. The existing text of the line serves as the line storage area until you press the <ENTER> key.

If you do not type a line number, (if you only press the <ENTER> key), the line after the current line (marked with an asterisk) is edited. If you do not want to change the current line, and the cursor is at the beginning or end of the line, press the <ENTER> key to accept the line as is.

IMPORTANT

If you press the <ENTER> key while the cursor is in the middle of the line, the remainder of the line is deleted.

EXAMPLE: Assume that the following file exists and is ready to edit (you may type this in to better follow the example):

```
1: This is a sample file
2: used to show
3: the editing of line
4: four.
```

To edit line 4, type after the *:

```
*4
```

and press <ENTER>

The contents of the line is displayed:

```
4:*four.
4:*
```

Now, using the editing keys, enter:

<Ins>number <F3>

The following appears:

```
4: number four.
5:*
```

A(PPEND)

APPEND

PURPOSE: Adds the specified number of lines from disk to the file being edited. The lines are added at the end of lines that are currently in memory.

FORMAT: [n]A

COMMENTS: This command is used only if the file being edited is too large to fit into memory. As many lines as possible are read into memory for editing when you start EDLIN.

To edit the remainder of the file that does not fit into memory, you must write the lines already edited to disk. You can then load unedited lines from disk into memory with the Append command. (Refer to the Write command description for information on how to write edited lines to disk.)

If you do not specify the number of lines to append, lines are appended to memory until available memory is three-fourths full. No action is taken if available memory is already three-fourths full.

The message "End of Input file" is displayed when the Append command has read the last line of the file into memory.

C(OPY)

PURPOSE: Inserts a range of lines ahead of a specified line number. You can copy the lines as many times as you want by using the count option.

FORMAT: [line1],[line2],line3[,count]C

WHERE: line1

Start of the text to be copied.

line2

End of the text to be copied.

line3

Line before which the text will be copied.

count

Number of times the lines are to be copied.

COMMENTS: If you do not specify a number in <count>, the EDLIN default is to copy the lines one time. If you omit the first or the second line number, and use a comma in its place, the default is the current line. The file is renumbered automatically after the copy, and the first of the copied lines becomes the current line number.

The line numbers must not overlap or an "Entry error" message appears. For example, 3,20,15C would result in an error message.

C(OPY)

EXAMPLES: Assume that the following file exists and is ready to edit:

```
1: This is a sample file
2: used to show copying lines.
3: See what happens when you use
4: the Copy command
5: (the C command)
6: to copy text in your file.
```

To copy this entire block of text, issue the following command:

```
1,6,7C
```

The result is:

```
 1: This is a sample file
 2: used to show copying lines.
 3: See what happens when you use
 4: the Copy command
 5: (the C command)
 6: to copy text in your file.
 7:*This is a sample file
 8: used to show copying lines.
 9: See what happens when you use
10: the Copy command
11: (the C command)
12: to copy text in your file.
```

To place the text within other text, the third line number should specify the line before which you want the copied text to appear. For example, if you want to copy lines and insert them within the following file:

```
 1: This is a sample file
 2: used to show copying lines.
 3: See what happens when you use
 4: the Copy command
 5: (the C command)
 6: to copy text in your file.
 7: You can also use COPY
 8: to copy lines of text
 9: to the middle of your file.
10: End of sample file.
```

The command 3,6,10C results in the following file:

```
 1: This is a sample file
 2: used to show copying lines.
 3: See what happens when you use
 4: the Copy command
 5: (the C command)
 6: to copy text in your file.
 7: You can also use COPY
 8: to copy lines of text
 9: to the middle of your file.
10: See what happens when you use
11: the Copy command
12: (the C command)
13: to copy text in your file.
14: End of sample file.
```

D(ELETE)

PURPOSE: Deletes a specified range of lines in a file.

FORMAT: [line1][,line2]D

COMMENTS: If you omit the first line number, that option defaults to the current line (the line with the asterisk next to the line number). If you omit the second line number, just the first line number is deleted. When you delete lines, the line immediately after the deleted section becomes the current line, and has the same line number that the first deleted line had before the deletion occurred.

EXAMPLES: Assume that the following file exists and is ready to edit:

```
    1: This is a sample file
    2: used to show dynamic line numbers.
    3: See what happens when you use
    4: Delete and Insert
    .
    .
    .
   25: (the D and I commands)
   26: to edit the text
   27: in your file.
```

To delete multiple lines (lines 5 through 24), type:

```
5,24D
```

The result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5:*(the D and I commands)
6: to edit the text
7: in your file.
```

To delete a single line, type only the line number desired. For example,

```
6D
```

Then, when you list the sample file with the L command, the result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6:*in your file.
```

Next, delete a range of lines from the following file:

```
1: This is a sample file
2: used to show dynamic line numbers.
3:*See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit the text
7: in your file.
```

To delete a range of lines beginning with the current line, and then to list the text again, type:

```
,6D
L
```

The result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3:*in your file.
```

Notice that the lines are automatically renumbered.

E(ND)

PURPOSE: Ends the editing session.

FORMAT: E

COMMENTS: This command saves the edited file on disk, renames the original input file <filename>.BAK, and then exits EDLIN. If the file is created during the editing session, no .BAK file is created.

The E command takes no options. Therefore, you cannot tell EDLIN on which drive to save the file. You must select the drive you want to save the file on when you start the editing session. If you do not select the drive when you start EDLIN, the file is saved on the disk in the default drive. You can, however, copy the file to a different drive using the COPY command.

Be sure that the disk contains enough free space for the entire file. If the disk does not contain enough free space, the write is cancelled, and the edited file is lost, although part of the file might be written out to the disk.

EXAMPLE: *E

After execution of the E command, the default drive prompt (for example, A>) is displayed.

I(NSERT)

PURPOSE: Inserts text immediately before the specified line number.

FORMAT: [line]I

COMMENTS: If you are creating a new file, you must give the I command before you can type (insert) text. Text begins with line number 1. Successive line numbers appear automatically each time you press the <ENTER> key.

EDLIN remains in insert mode until you enter <Ctrl-Break> or <F6>. When the insert is completed and insert mode has been exited, the line immediately following the inserted lines becomes the current line. All line numbers following the inserted section are increased by the number of lines inserted.

If you do not specify the line number, the lines are inserted immediately before the current line. If the line number is any number larger than the last number, or if you specify a pound sign (#) in place of 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.

EXAMPLES: Assume that the following file exists and is ready to edit:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit the text
7: in your file.
```

To insert text before a specific line that is not the current line, type the line number. For example, type

```
*7I
```

The result is:

```
7:
```

Now, type the new text for line 7:

```
7: and renumber lines
```

Then, to end the insertion, press:

8: <F6> and <ENTER>

Now enter L to list the file. The result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit the text
7: and renumber lines
8:*in your file.
```

To insert lines immediately before the current line, type:

```
I
```

The result is:

```
8:_
```

Now, type the following text:

```
8: so they are consecutive
9: <F6><ENTER>
```

Now, when you list the file, the result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit the text
7: and renumber lines
8: so they are consecutive
9:*in your file.
```

To append new lines to the end of the file, type:

```
#I
```

This produces the following:

```
10:
```

Now, type the following new lines:

```
10: The insert command can place new lines
11: in the file; there's no problem
12: because the line numbers are dynamic;
13: they'll go all the way to 65529.
```

To end the insertion, press <F6> and <ENTER> on line 14.

```
14: <F6>
```

I(NSERT)

The new lines appear at the end of all previous lines in the file. Now type the list command, L:

The result is:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit the text
 7: and renumber lines
 8: so they are consecutive
 9: in your file.
10: The insert command can place new lines
11: in the file; there's no problem
12: because the line numbers are dynamic;
13: they'll go all the way to 65529.
```

L(IST)

PURPOSE: Lists a range of lines, including the two lines specified.

FORMAT: [line1][,line2]L

COMMENTS: Default values are provided if you omit either one or both of the options. If you omit the first option, as in:

```
,line2L
```

the display starts 11 lines before the current line, and ends with the specified line. The beginning comma is required to indicate the omitted first option.

NOTE: If the line you specify <line2> is more than 11 lines before the current line, the display is the same as if you omitted both options.

If you omit the second option, as in:

```
line1L
```

23 lines are displayed, starting with the specified line number.

If you omit both parameters, as in:

```
L
```

23 lines are displayed: the 11 lines before the current line, the current line, and the 11 lines after the current line. If there are fewer than 11 lines before the current line, more than 11 lines after the current line are displayed to make a total of 23 lines.

L(IST

EXAMPLES: Assume that the following file exists and is ready to edit:

```
    1: This is a sample file
    2: used to show dynamic line numbers.
    3: See what happens when you use
    4: Delete and Insert
    5: (the D and I commands)
    .
    .
    .
   15:*The current line contains an asterisk.
    .
    .
    .
   26: to edit the text
   27: in your file.
```

To list a range of lines without reference to the current line, type:

```
  2,5L
```

The result is:

```
    2: used to show dynamic line numbers.
    3: See what happens when you use
    4: Delete and Insert
    5: (the D and I commands)
```

To list a range of lines beginning with the current line, type:

```
  ,26L
```

The result is:

```
   15:*The current line contains an asterisk.
    .
    .
    .
   26: to edit the text
```

To list a range of 23 lines centered around the current line, type:

```
L
```

The result is:

```
 4: Delete and Insert
 5: (the D and I command)
 .
 .
 .
13: The current line is listed in the middle of
the range.
14: The current line remains unchanged by the L
command.
15:*The current line contains an asterisk.
 .
 .
 .
26: to edit the text
```

M(OVE)

PURPOSE: Moves a range of text to the line specified.

FORMAT: [line1],[line2],line3M

COMMENTS: Use the Move command to move a block of text (from the first line number (line1) through the second (line2)) to another location in the file (in front of line3). The lines are renumbered according to the direction of the move. In other words, the lines following the section to be moved are moved up accordingly. For example:

```
,+25,100M
```

relocates the block of text from the current line plus 25 lines in front of line 100. If the line numbers overlap, EDLIN displays an "Entry error" message.

To move lines 20-30 to line 100, type:

```
20,30,100M
```

P(AGE)

PURPOSE: Displays a block of lines, and moves current line marking to last line displayed.

FORMAT: [line1][,line2]P

COMMENTS: If you omit the first line (line1), that number defaults to the current line plus one. If you omit the second line (line2), 23 lines are listed. The last line displayed becomes the new current line and is marked with an asterisk.

Successive P commands, with no line numbers specified, page through the file 23 lines at a time.

Q(UIT)

PURPOSE: Quits the editing session, does not save any editing changes, and exits to DOS.

FORMAT: Q

COMMENTS: EDLIN prompts you to make sure you don't want to save the changes.

Type Y if you want to quit the editing session. No editing changes are saved and no .BAK file is created. Refer to the End command description for information about the .BAK file.

Type N or any other character if you want to continue the editing session.

NOTE: When you start EDLIN, it erases any previous copy of the file with an extension of .BAK to make room to save the new copy. If you reply Y to the "(Y/N)?" message, your previous backup copy no longer exists.

EXAMPLE:
```
Q
Abort edit (Y/N)? Y
A>_
```

R(EPLACE)

PURPOSE: Replaces all occurrences of a string of text in the specified range with a different string of text.

FORMAT: [line1][,line2][?]R[string1] [<F6>string2]

COMMENTS: As each occurrence of <string1> is found, it is replaced by <string2>. Each line in which a replacement occurs is displayed. If a line contains two or more replacements of <string1> with <string2>, the line is displayed once for each occurrence. When all occurrences of <string1> in the specified range are replaced by <string2>, the R command terminates and the EDLIN prompt (asterisk) appears.

If a second string is to be given as a replacement, then you must separate <string1> from <string2> with <F6> (or <Ctrl-Z). <string2> ends with <ENTER>.

If you omit <string1>, Replace takes the old <string1> as its value. If there is no old <string1> (that is, this is the first Search or Replace done), the replacement process is terminated immediately. If you omit <string2>, then <string1> may end with <ENTER>. If you omit the first line number in the range argument by using a comma (as in ",<line2>"), <line1> defaults to the line after the current line. If you omit the second line number (line2) (as in "line1" or "line1,"), the second line defaults to #. Remember that # indicates the line after the last line of the file.

If <string1> ends with <F6> and there is no <string2>, <string2> is taken as an empty string and becomes the new replace string. For example,

R<string1><F6>

deletes occurrences of <string1>, but

R<string1><ENTER>

replaces the new <string1> with the old <string2>.

R(EPLACE)

R<ENTER>

replaces the old <string1> with the old <string2>.

Note that "old" here refers to a previous string specified in either a Search or a Replace command.

If you use the question mark (?) option, the Replace command stops at each line with a string that matches <string1>, displays the line with <string2> in place, and then displays the prompt "O.K.?". If you press Y or the <ENTER> key, <string2> replaces <string1>, and the next occurrence of <string1> is found. Again, the "O.K.?" prompt is displayed. This process continues until the end of the range, or until the end of the file. After the last occurrence of <string1> is found, the EDLIN prompt appears.

If you press any key except Y or <ENTER> after the "O.K.?" prompt, the <string1> is left as it was in the line, and Replace goes to the next occurrence of <string1>. In this way, only the desired <string1> is replaced, and you can prevent unwanted substitutions.

EXAMPLES: Assume that the following file exists and is ready for editing:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit the text
 7: in your file.
 8: The insert command can place new lines
 9: in the file; there's no problem
10: because the line numbers are dynamic;
11: they'll go all the way to 65520.
```

To replace all occurrences of <string1> (in this example, "and") with <string2> ("or") in a specified range (lines 2 thru 12), type:

```
2,12 Rand<F6>or<ENTER>
```

The result is:

```
4: Delete or Insert
5: (the D or I commands)
8: The insert command can place new lines
```

Note that in the replacements, some unwanted substitutions have occurred. To avoid these, and to confirm each replacement, use the original file with a slightly different command. In the next example, to replace only certain occurrences of the first string with the second string, type:

```
2,12?Rand<Ctrl-Z>or<ENTER>
```

The result is:

```
4: Delete or Insert
O.K.? Y
5: (the D or I commands)
O.K.? Y
5: (the D or I command)
O.K.? N
8: The insert command can place new lines
O.K.? N
*
```

Now, type the List command (L) to see the result of all these changes:

```
.
4: Delete or Insert
5: (the D or I commands)
.
.
.
8: The insert command can place new lines
.
```

S(EARCH)

PURPOSE: Searches the specified range of lines for a specified string of text.

FORMAT: [line1][,line2][?]S string

COMMENTS: You must press <ENTER> to end the string. The first line that matches string is displayed and becomes the current line. If you do not specify the question mark option, the Search command terminates when a match is found. If no line contains a match for string, the message "Not found" is displayed.

If you include the question mark option (?) in the command, EDLIN displays the first line with a matching string, and prompts you with the message "O.K.?". Press either the Y or <ENTER> key to terminate the search and make the line the current line. Press any other key to continue the search until another match is found, or until all lines are searched (and the "Not found" message is displayed).

If you omit the first line (line1) (as in ",<line2>S<string>"), <line1> defaults to the line after the current line. If you omit <line2> (as in "<line1>S<string>" or "<line1>,S<string>"), <line2> defaults to the pound sign (#) (the line after the last line of the file), which is the same as "<line1>,#S<string>". If you omit <string>, Search takes the old string if there is one. ("old" here refers to a string specified in a previous Search or Replace command.) If there is not an old string, the command terminates.

EXAMPLES: Assume that the following file exists and is ready for editing:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit the text
 7: in your file.
 8: The insert command can place new lines
 9: in the file: there's no problem
10: because the line numbers are dynamic;
11:*they'll go all the way to 65529.
```

To search for the first occurrence of the string "and," type

```
2,12 Sand
```

The following line is displayed:

```
4: Delete and Insert
```

to get the "and" in line 5, modify the search command by typing:

```
<Del><F3>
```

The search then continues from the line after the current line (line 4), since no first line was given. The result is:

```
5: (the D and I commands)
```

To search through several occurrences of a string until the correct string is found, type:

```
1, ?Sand
```

The result is:

```
4: Delete and Insert
O.K.?
```

If you press any key except Y or <ENTER>, the search continues, so type N:

```
O.K.? N
```

Continue:

```
    5: (the D and I commands)
  O.K.?
```

Now press Y to terminate the search:

```
O.K? Y
*
```

To search for string XYZ without the verification (O.K.?), type:

```
SXYZ
```

EDLIN finds a match and continues to search for the same string when you enter the S command:

```
S
```

When you enter the S command, EDLIN continues to search until there are no more occurrences, which brings about the message.

```
Not found
```

Note that <string> defaults to any string specified by a previous Replace or Search command.

T(RANSFER)

PURPOSE: Inserts (merges) the contents of filename[.ext] into the file currently being edited at <line>. If you omit <line>, the current line is used.

FORMAT: [line]T[d:]filename[.ext]

COMMENTS: This command is useful if you want to put the contents of a file into another file or into text you are typing. The transferred text is inserted ahead of the line number specified by <line>, and the lines are renumbered after the inserted numbers.

W(RITE)

PURPOSE: Writes a specified number of lines to disk from the lines that are being edited in memory. Lines are written to disk beginning with line number 1.

FORMAT: [n]W

COMMENTS: This command is useful only if the file you are editing is too large to fit into memory. When you start EDLIN, it reads lines into memory until memory is three-fourths full.

To edit the remainder of your file, you must write edited lines in memory to disk. You can then load additional lines from disk into memory by using the Append command.

NOTE: If you do not specify the number of lines, lines are written to disk until memory is one-fourth full. No action is taken if available memory is already less than one-fourth full. All lines are renumbered, so that the first remaining line becomes line 1.

Section 6
LINK PROGRAM (LINK)

In this section:

GENERAL INFORMATION

This section tells you how to use LINK. You should read the entire section before you use LINK.

NOTE: If you are not going to compile and link programs, you do not need to read this section.

LINK is a program that performs the following functions:

- Combines one or more separately produced object modules into one relocatable load module (a program you can run).
- Searches library files for definitions of unresolved external references.
- Resolves external cross-references.
- Produces a listing that shows both the resolution of external references and error messages.

Program Overview

When you write a program, you write it in source code. This source code is run through a compiler or an assembler, which produces object modules. The object modules must be passed through the link process to produce an executable module that the computer can understand directly.

You may wish to link (combine) several programs and run them together. Any of your programs can refer to one or more symbols that are defined in other modules. These references are called external references.

LINK combines several object modules into one relocatable load module or Run file (called an .EXE or Executable file). As it combines modules, LINK makes sure that all external references between object modules are defined. LINK can search several library files for definitions of any external references that are not defined in the object modules.

LINK also gives you the option of producing a List file that shows external references resolved, and it displays all error messages.

LINK uses available memory as much as possible. When available memory is exhausted, LINK creates a temporary disk file named VM.TMP.

Figure 6-1 illustrates the various parts of the LINK operation.

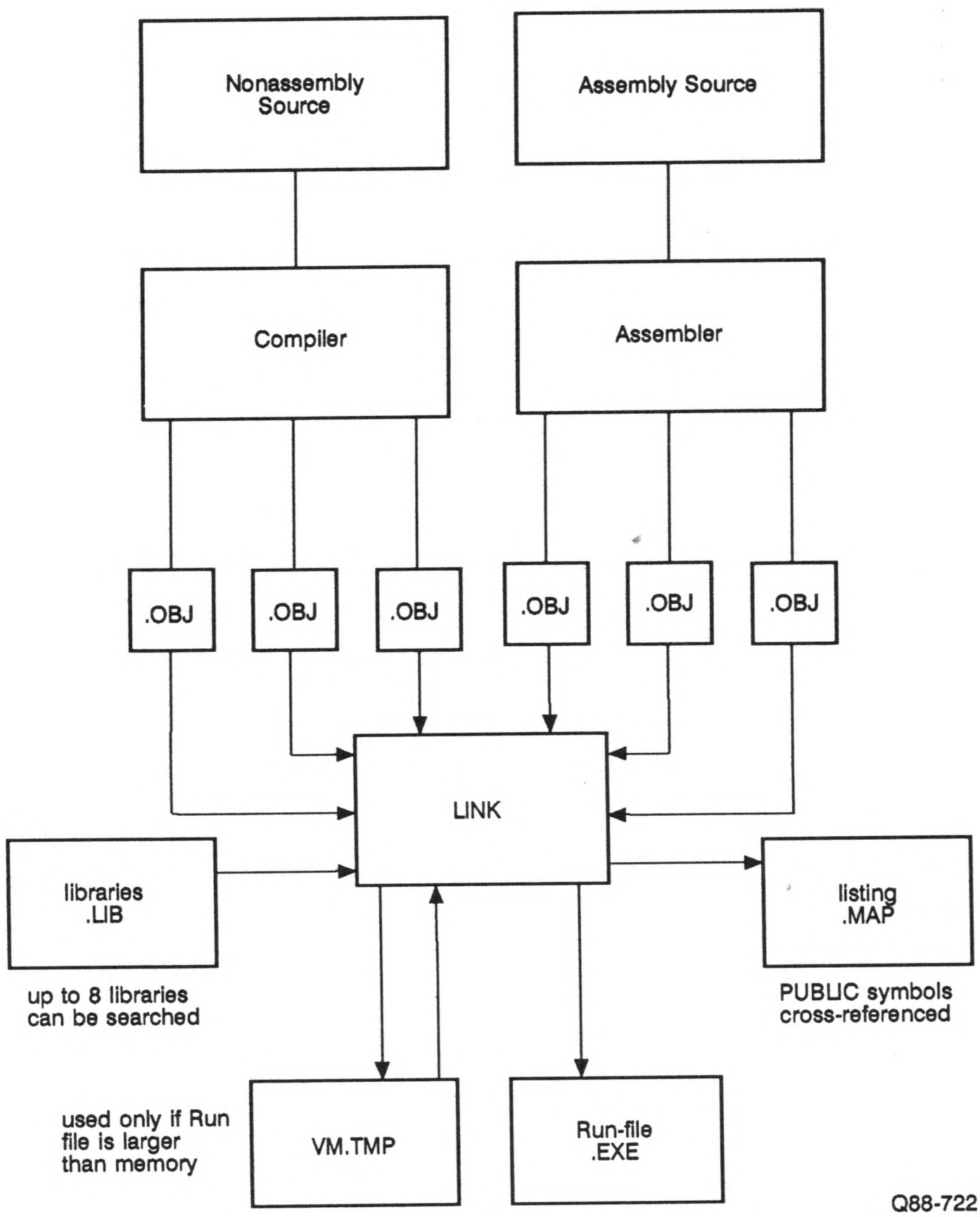


Figure 6-1. The LINK Operation

Definitions

Some of the terms used in this section are explained below to help you understand how LINK works. Generally, if you are linking object modules compiled from BASIC, Pascal, or another high-level language, you do not need to know these terms. If you are writing and assembling programs in Assembly language, however, you need to understand LINK and the definitions described in this section.

In MS-DOS, memory can be divided into segments, groups, and classes. Figure 6-2 illustrates these concepts.

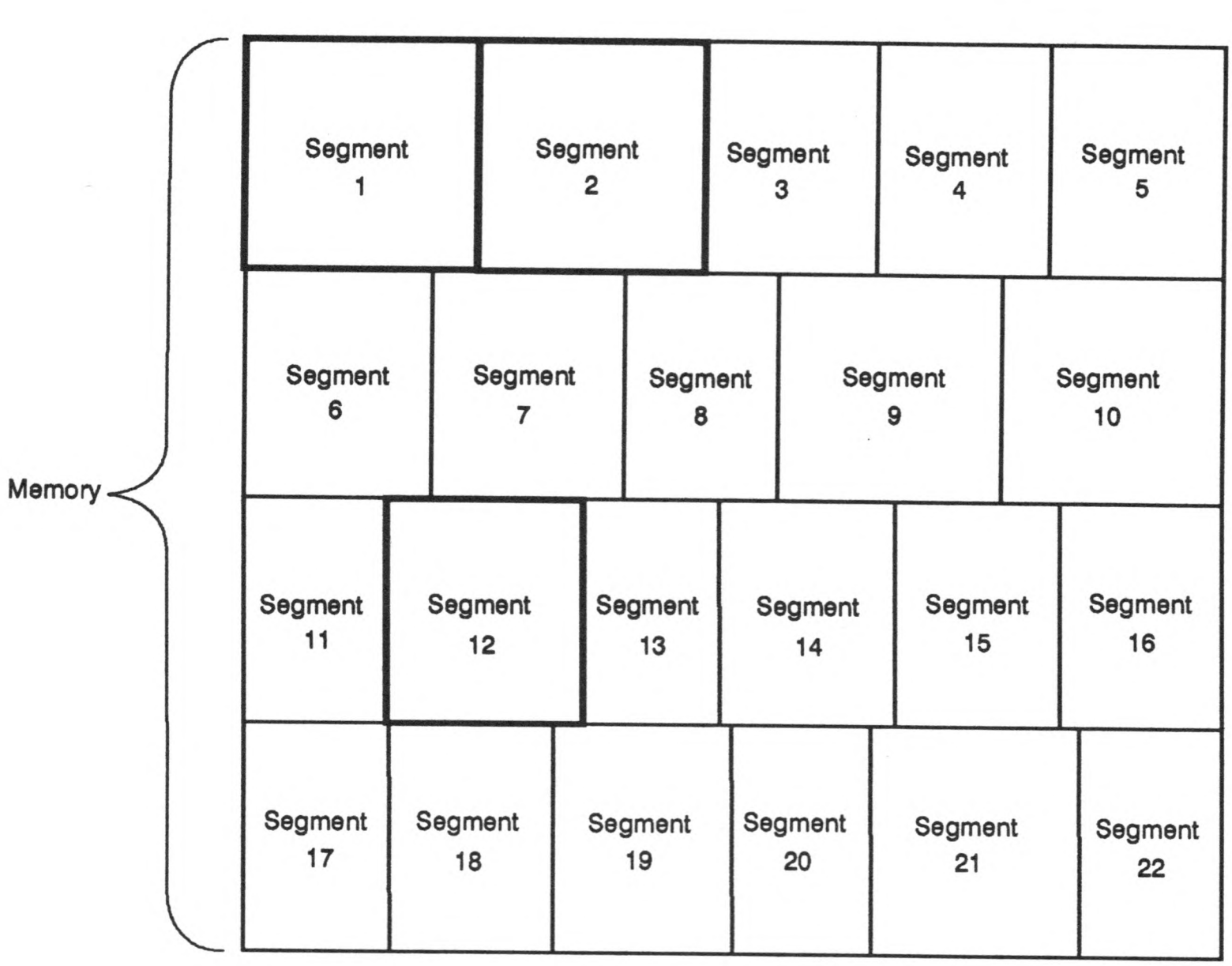


Figure 6-2. How Memory is Divided

Segment

A segment is a piece of a program that represents a logical function, module, or structure. It can correspond to a physical segment that is a contiguous area of memory as large as 64 KB in length. A segment has an alignment type of: BYTE, WORD, PARA, or PAGE; and is aligned by LINK on a 1, 2, 16, or 1024 byte memory boundary, respectively.

Group

A group is a collection of dissimilar segments that fit together within a 64 KB physical segment of memory. The segments are named to the group by the assembler or compiler. A program can consist of one or more groups. The group is used for addressing dissimilar software segments in the same physical segment of memory.

Class

A class is a collection of similar segments assigned a name. The naming of segments to a class affects the order and relative placement of segments in memory. The class name is specified by the assembler or compiler. All portions assigned to the same class name are loaded into memory continuously. Otherwise, each module is loaded in the order it is entered into the LINK program.

Each segment has a segment name and a class name. LINK loads all segments into memory by class name from the first segment encountered to the last. All segments assigned to the same class are loaded into memory contiguously. Group names do not affect the order in which segments are loaded. LINK also uses combine types (public, stack, memory, common and private) when combining segments. All public, stack and memory segments having the same name and class, respectively, are loaded contiguously. Stack segments cause LINK to copy an initial stack pointer to the executable file. This pointer value is the offset to the end of the first segment encountered. For common segments, Link combines the segments at the same common starting address (overlaps the segments). For private segments (the default, if no other combine type is specified), no combination of segments occurs.

During processing, LINK references segments by their addresses in memory (where they are located). LINK does this by finding the frame number and offset into the segment. The frame number specifies first paragraph in memory containing a byte of the segment and is the first five hexadecimal digits of the "start" address (specified for the segment as contained in the map file from the LINK).

The segments do not need to be contiguous to form a group (Figure 6-2). The address of any group is the lowest address of the segments in that group. At link time, LINK analyzes the groups and then references the segments by the address in memory of that group. A program can consist of one or more groups.

If you are writing in Assembly language, you can assign the group and class names in your program. In Assembly language programs, you can control the segment loading order by first linking a dummy application that identifies all classes of segments in the desired order of loading and contains no programming code. In high-level languages (BASIC, COBOL, FORTRAN, Pascal), the naming is done automatically by the compiler.

Files That LINK Uses

LINK works with one or more input files, produces one or two output files, can create a temporary disk file, and can be directed to search up to eight library files.

For each type of file, you can give a four-part file specification. The format for LINK file specifications is the same as that of a disk file:

```
[d:][pathname]filename[.ext]
```

d:

Drive designation. The colon is always required as part of the drive designation.

pathname

Directory path in which the file is located.

filename

Any legal file name of one to eight characters.

.ext

One- to three-character extension to the file name. The period is always required as part of the extension.

Input File Extensions

If no file name extensions are given in the input (object) file specifications, LINK appends the following extensions by default:

.OBJ	Object
.LIB	Library

Output File Extensions

LINK appends the following default extensions to the output (Run and List) files:

.EXE	Run (cannot be overridden)
.MAP	List (can be overridden)

VM.TMP (Temporary) File

LINK uses available memory for the link session. If the files to be linked create an output file that exceeds available memory, LINK creates a temporary file, names it VM.TMP, and puts it on the disk in the default drive. If LINK creates VM.TMP, it displays the message:

VM.TMP has been created.
Do not change disk in drive <d:>

When this message is displayed, you must not remove the disk from the default drive until the link session ends. If you remove the disk, the operation of LINK will be unpredictable, and LINK might display the error message:

Unexpected end of file on VM.TMP

The contents of VM.TMP are written to the file named following the Run File: prompt. VM.TMP is a working file only and is deleted at the end of the linking session.

IMPORTANT

Do not use VM.TMP as a file name for any file. If you have a file named VM.TMP on the default drive and LINK requires the VM.TMP file, LINK deletes the VM.TMP already on the disk and creates a new VM.TMP. Thus, the contents of the previous VM.TMP file will be lost.

USING LINK

Starting LINK

LINK requires two types of input: a command to start LINK and responses to command prompts. In addition, six options control LINK features. Usually, you type all the commands to LINK on the keyboard. As an option, answers to the command prompts and any options can be contained in a response file. Three special command characters [plus sign (+), semicolon (;), and at-sign (@)] can be used to modify the sequence of the commands you give to LINK.

You can start LINK in any of three ways:

- LINK
- LINK filenames [/options]
- LINK @filespec

The first method is to type the commands in response to individual prompts. In the second method, you type all commands on the line used to start LINK. To start LINK by the third method, you must create a response file that contains all the necessary LINK commands and tell LINK where that file is when you start LINK. This method is best used if you have many source programs and anticipate extensive relinking.

Method 1: Prompts

To start LINK with method 1, type:

```
LINK
```

when the MS-DOS diskette is loaded.

LINK is loaded into memory, and then LINK displays four text prompts that appear one at a time. You answer the prompts to tell LINK to perform specific tasks.

At the end of each line, you can type one or more options, preceded by the parameter character (in this case, a slash (/)).

The command prompts are summarized in the following table, and are described in more detail under "Command Prompts," in this section.

Prompt	Responses
Object Modules [.OBJ]:	List .OBJ files to be linked. They must be separated by spaces or plus signs (+). If a plus sign is the last character typed, the prompt reappears. There is no default; a response is required.
Run File [filename.EXE]:	Give file name for executable object code. The default is first-object-filename.EXE. (You cannot change the output file name extension.)
List File [NUL.MAP]:	Give file name if you want a listing. The default is no listing (NUL).
Libraries [.LIB]:	List file names to be searched, separated by spaces or plus signs (+). If a plus sign is the last character typed, the prompt reappears. The default is an automatic library search using the .LIB extension.

Method 2: Command Line

To start LINK using method 2, type all commands on one line. The entries following LINK are responses to the command prompts. The entry fields for the different prompts must be separated by commas. Use the following format:

```
LINK object-list,runfile,listfile,library-list
[/option...]
```

object-list

List of object modules, separated by plus signs.

runfile

Name of the file to receive the executable output.

listfile

Name of the file to receive the listing.

library-list

List of library modules to be searched.

/option

Refers to processing choices (refer to "Link Options", later in this section) that can be placed following any of the response entries (just before any of the commas or after the library-list, as shown).

To select the default for a field, simply type a second comma with no spaces between the two commas:

```
LINK PROCESS+TEXT+TABLE+CALC/P/M,,
PROCLIST,COBLIB.LIB
```

This command causes LINK to be loaded, followed by the object modules PROCESS.OBJ, TEXT.OBJ, TABLE.OBJ, and CALC.OBJ. LINK then pauses (as a result of using /P option). LINK links the object modules when you press any key, and produces a universal symbol map (the /M option); defaults to PROCESS.EXE run file; creates a list file named PROCLIST.MAP; and searches the library file COBLIB.LIB.

Method 3: Response File

To start LINK with method 3, type:

```
LINK @[d:][pathname]filename[.ext]
```

where the filespec is the name of a response file. A response file contains answers to the LINK prompts (shown in method 1) and can also contain any of the options. When naming a response file, the use of file name extensions is optional. Method 3 permits the command that starts LINK to be entered from the keyboard or within a batch file without requiring you to take any further action.

To use this option, you must create a response file containing several lines of text, each of which is the response to a LINK prompt. The responses must be in the same order as the LINK prompts discussed in method 1. If desired, a long response to the "Object Modules:" or "Libraries:" prompt can be typed on several lines by using a plus sign (+) to continue the same response onto the next line.

Use options and command characters in the response file the same way as they are used for responses typed on the terminal keyboard.

When the LINK session begins, each prompt is displayed in order with the responses from the response file. If the response file does not contain answers for all the prompts (in the form of file names, the semicolon command character, or <ENTER>), LINK displays the prompt that does not have a response and then waits for you to type a legal response. When a legal response is typed, LINK continues the link session. Consider the following example:

```
PROCESS TEXT TABLE CALC
/PAUSE/MAP
PROCLIST
COBLIB.LIB
```

This response file tells LINK to load the four object modules named PROCESS, TEXT, TABLE, and CALC. Note that the second response line consists only of the /PAUSE and /MAP options. Thus the default value for Run File (first object file name +.EXE, in this case PROCESS.EXE) has effectively been chosen. LINK pauses before producing a public symbol map to permit you to swap disks. (Refer to the discussion of /PAUSE under "Link Options" in this section before using this feature.) When you press <ENTER>, the output files are named PROCESS.EXE and PROCLIST.MAP. LINK searches the library file COBLIB.LIB and uses the default setting for the options.

Command Characters

LINK provides three command characters:

Plus Sign (+)

Use the plus sign (+) to separate entries and to extend the current line in response to the "Object Modules:" and "Libraries:" prompts. (A space can be used to separate object module or library names.) To type a large number of responses (each of which can be very long), type a plus sign and an <ENTER> at the end of the line to extend it. If the plus sign and <ENTER> are the last entries following either of these two prompts, LINK prompts you for more module or library names. When the "Object Modules:" or "Libraries:" prompt appears again, continue to type responses. When all the modules to be linked and libraries to be searched are listed be sure the response line ends with a module name and an <ENTER> and not a plus sign and <ENTER>.

EXAMPLE:
```
Object Modules [.OBJ]: PROCESS TEXT TABLE CALC+
Object Modules [.OBJ]: READDATA+VALIDATE+
Object Modules [.OBJ]: BOLDPRNT+
Object Modules [.OBJ]: PIECHART
```

Semicolon (;)

To select default responses to the remaining prompts, use a semicolon (;) followed immediately by an <ENTER> at any time after the prompt for Run File. This feature saves time and overrides the need to press a series of <ENTER> keys.

NOTE: Once the semicolon has been typed and entered (by pressing the <ENTER> key), you can no longer respond to any of the prompts for that link session. To skip individual prompts, use the <ENTER> key.

EXAMPLE:
```
Object Modules [.OBJ]: PROCESS TEXT TABLE
CALC<ENTER>
Run File [PROCESS.EXE]: ;<ENTER>
```

No other prompts appear, and LINK uses the default values (including PROCESS.MAP for the list file).

<Ctrl-Break>

Use the <Ctrl-Break> keys to cancel the link session at any time. If you type an erroneous response, such as the wrong file name or an incorrectly spelled file name, you must press <Ctrl-Break> to exit LINK, and then restart LINK. However, if you typed the error but did not press the <ENTER> key, you can delete the erroneous characters with the backspace key, but for that line only.

Command Prompts

LINK prompts the user for the names of Object, Run, and List files, and for Libraries. The prompts are listed in order of appearance. The default response is shown in square brackets [ ] following the prompt, for prompts that can default to preset responses. The "Object Modules:" prompt, however, has no preset file name response and requires you to type a file name.

Object Modules [.OBJ]:

Type a list of the object modules to be linked. LINK assumes by default that the file name extension is .OBJ. If an object module has any other file name extension, the extension must be given. Otherwise, the extension can be omitted.

Modules must be separated by plus signs (+) or spaces.

Remember that LINK loads segments into classes in the order encountered. You can use this information to set the order in which the object modules are read by LINK.

Run File [First-Object-filename.EXE]:

Typing a file name creates a file for storing the Run (executable) file that results from the link session. All Run files receive the file name extension.EXE, even if you specify an extension other than .EXE.

If no response is typed to the "Run File:" prompt, LINK uses the first file name typed in response to the "Object Modules:" prompt as the RUN file name.

EXAMPLE: Run File [PROCESS.EXE]: B:PAYROLL/P

This response directs LINK to create the Run file PAYROLL.EXE on drive B:. The /P causes link to pause, which allows you to insert a new disk to receive the Run file.

List File [NUL.MAP]:

The list file contains an entry for each segment in the input (object) modules. Each entry shows the addressing in the Run file. This file is very useful when using the DEBUG Utility.

The default response is no listing (NUL).

Libraries [.LIB]:

The valid responses are up to eight library file names or simply <ENTER>. Library files must have been created by a library utility. LINK assumes by default that the file name extension is .LIB for library files.

Library file names must be separated by spaces or plus signs (+).

LINK searches library files in the order listed to resolve external references. When it finds the module that defines the external symbol, LINK processes that module as another object module.

If LINK cannot find a library file on the disk(s) currently in the disk drive(s), it displays the message:

```
Cannot find library <d:library-name>
Enter new drive letter:
```

Press the letter for the drive designation (for example, B).

LINK Options

LINK has the following options:

/C[PARMAXALLOC]	Maximum Allocation Space
/DO[SSEG]	Use DOS Segment Ordering
/DS[ALLOCATE]	Data Group Allocation
/HELP	Help
/H[IGH]	High Load
/L[INENUMBERS]	Line Number
/M[AP]	Public Symbol Map
/NOD[EFAULTLIBRARYSEARCH]	Default Library Override
/NOG[ROUPASSOCIATION]	Group Association Override
/NOI[GNORECASE]	Case Sensitivity in Name
/O[VERLAYINTERRUPT]	Overlay Interrupt
/P[AUSE]	Pause During Linking
/SE[GMENTS]	Segment Number Maximum
/ST[ACK]	Stack Size

The LINK options control various LINK functions. You must type options at the end of a prompt response, regardless of the method you use to start LINK. You can group options at the end of any response, or scatter them at the end of several. If you type more than one option at the end of one response, you must precede each option with a slash (/).

You can abbreviate all LINK option names, as shown in the preceding listing (the characters inside the brackets ([]) are optional). The only restriction is that an option name abbreviation must be sequential from the first letter through the last typed; no gaps or transpositions are allowed.

/C[PARMAXALLOC]:<number>

The /C[PARMAXALLOC] option sets the maximum number of 16-byte paragraphs needed by the program when it is loaded into memory. This number is used by the operating system when allocating space for the program prior to loading it. The number can be any integer value in the range from 1 to 65,535. It must be a decimal, octal, or hexadecimal number. Octal numbers must begin with a zero. Hexadecimal values must begin with 0x.

LINK normally sets the maximum number of paragraphs to 65,535. Since this represents all of memory, the operating system always denies the request and allocates the largest contiguous block of memory it can find. If the /C[PARMAXALLOC] option is used, the operating system will allocate no more space than given by this option. This means any additional space in memory is free for other purposes.

If number is less than the minimum number of paragraphs needed by the program, LINK ignores your request and sets the maximum value equal to the minimum. The minimum number of paragraphs needed by a program is never less than the number of paragraphs of code and data in the program.

EXAMPLES: LINK file.obj/C:15,file.exe,,;

Sets the maximum allocation to 15 paragraphs.

LINK moda+modb,run/CPARMAXALLOC:0xff,ab.map,;

Sets the maximum allocation to 255 (FFH) paragraphs.

LINK startup+file,file/C:030,,;

Sets the maximum allocation to 24 (30 octal) paragraphs.

/DO[SSEG]

The /DOSSEG option causes LINK to arrange all segments in the executable file according to the DOS segment ordering convention. This convention has the following rules:

1. All segments having the class name, "CODE," are placed at the beginning of the executable file.

2. Any segments that do not belong to the group named "DGROUP" are placed immediately after the "CODE" segments.

3. All segments belonging to DGROUP are placed at the end of the file.

EXAMPLE: LINK start+test/DOSSEG,text,,math+common

Causes LINK to create an executable file, named "file.exe," whose segments are arranged according to the DOS segment ordering convention. The segments in the object files "start.obj" and "test.obj," and any segments copied from the libraries "math.lib" and "common.lib" are arranged in the order specified above.

/DS[ALLOCATE]

The DSALLOCATE option directs LINK to reverse its normal processing when assigning addresses to items belonging to the group named "DGROUP." Normally, LINK assigns the offset 0000H to the lowest byte in a group. If /DSALLOCATE is given, LINK assigns the offset FFFFH to the highest byte in the group. The result is data that appears to be loaded as high as possible in the memory segment containing DGROUP.

The /DSALLOCATE option is typically used with the /HIGH option to take advantage of unused memory before the start of the program. LINK assumes that all free bytes in DGROUP occupy the memory immediately before the program. To use the group, a segment register must be set to the start address of DGROUP.

EXAMPLE: LINK startup+file/HIGH/DSALLOCATE,file,file,;

Directs LINK to place the program as high in memory as possible, then adjust the offset of all data items in DGROUP so that they are loaded as high as possible within the group.

NOTE: Your application program may dynamically allocate up to 64 KB (or the actual amount of memory available) less the amount allocated within DGROUP.

/HELP

The /HELP option causes link to write a list of available options to the screen. If you ever need a reminder of the available options, you may find this list convenient. You should not give a file name when using the /HELP option.

Minimum abbreviation /HE

EXAMPLE: LINK /HELP

/H[IGH]

The /HIGH option sets the program's (Run file) starting address to the highest possible address in free memory. This option actually causes LINK to add information to the executable file that makes the operating system load the program as high as possible.

If /HIGH is not given, the program's starting address is set as low as possible in memory.

EXAMPLE: LINK startup+file/HIGH,file,file,;

Sets the starting address of the program in "file.exe" to the highest possible address in free memory.

IMPORTANT

Do not use the /HIGH option with Pascal or FORTRAN programs.

/L[INENUMBERS]

The /LINENNUMBERS option directs LINK to list the starting address of each program source line. The starting address is actually the address of the instructions that make up the corresponding source line. LINK copies this information to the map file where you can use it for program debugging.

LINK carries out the line numbering only if you give a map file name in the LINK command line, and only if the given object file has line number information. Line numbering is available only in high-level language compilers. If an object file has no line number information, LINK ignores the /LINENUMBERS option.

NOTE: If you do not specify a map file in a LINK command, you can use the /LINENUMBERS option to force LINK to create a map file by placing the option at or before the "List file" prompt. LINK gives the forced map file the same file name as the first object file specified in the command and the default extension .MAP.

EXAMPLE: LINK file.obj/LINENUMBERS,file.exe,file.map,;

Causes the line number information in the object file "file.obj" to be copied to the map file "file.map."

NOTE: Not all compilers produce object modules that contain line number information. In these cases, of course, LINK cannot include line numbers.

/M[AP]

The /MAP] option causes LINK to produce a listing of all public symbols declared in your program. This list is copied to the map file created by LINK.

NOTE: If you do not specify a map file in a LINK command, you can use the /MAP option to force LINK to create a map file by placing the option at or before the "List file" prompt. LINK gives the forced map file the same file name as the first object file specified in the command and the default extension .MAP.

EXAMPLE: LINK file.obj,file.exe,file.map/MAP,;

Creates a map of all public symbols in the file file.obj.

The symbols are listed alphabetically. For each symbol, LINK lists its value and its segment offset location in the Run file. The symbols are listed at the end of the List file.

/NOD[EFAULTLIBRARYSEARCH]

The /NODEFAULTLIBRARYSEARCH option directs LINK to ignore any library names it may find in an object file. A high-level language compiler may add a library name to an object file to ensure that a default set of libraries are linked with the program. Using this option overrides these default libraries and lets you explicitly name the libraries you want on the LINK command line.

EXAMPLE: LINK startup+file/NOD,file.exe,,\lib\math.lib

Links the object files startup.obj and file.obj with routines from the library \lib\math.lib. Any default libraries that may have been named in startup.obj or file.obj are ignored.

/NOG[ROUPASSOCIATION]

The /NOGROUPASSOCIATION option directs LINK to ignore group associations when assigning addresses to data and code items.

NOTE: Use of this option is not recommended.

/NOI[GNORECASE]

The /NOIGNORECASE option directs LINK to treat uppercase and lowercase letters in symbol names as distinct letters. Normally, LINK considers uppercase and lowercase letters to be identical, treating the names "TWO," "two," and "Two" as the same symbol. Using /NOIGNORECASE causes the LINK to treat these names as unique symbols.

The /NOIGNORECASE option is typically used with object files created by high-level language compilers. Some compilers treat uppercase and lowercase letters as distinct letters and assume that LINK will too.

EXAMPLE: LINK file.obj/NOI,file.exe,file.map,\lib\Slibc.lib;

Causes LINK to treat uppercase and lowercase letters in symbol names as distinct letters. The object file "file.obj" is linked with routines from the standard C language library "\lib\Slibc.lib." The C language expects uppercase and lowercase letters to be treated separately.

/O[VERLAYINTERRUPT]:number

The /OVERLAYINTERRUPT option sets the interrupt number of the overlay loading routine to number. This option overrides the normal overlay interrupt number (03FH).

The number can be any integer value in the range from 0 to 255. It must be a decimal, octal, or hexadecimal number. Octal numbers must have a leading zero. Hexadecimal numbers must start with 0x.

NOTE: Using interrupt numbers that conflict with the standard DOS interrupts is not recommended.

EXAMPLES: LINK file.obj/O:255,file.exe,,;

Sets the overlay interrupt number to 255.

LINK mode+modb,run/OVERLAY:0xff,ab.map,;

Sets the overlay interrupt number 255 (FFH).

LINK startup+file,file/O:0377,,;

Sets the overlay interrupt to 255 (377 octal).

/P[AUSE]

The /PAUSE option causes LINK to pause before writing the executable file to disk. This option allows you to swap disks before LINK outputs the executable (.EXE) file.

If the /PAUSE option is given, LINK displays the following message before creating the run file:

```
About to generate .EXE file
Change diskette in drive X and press <ENTER>
```

LINK resumes processing when you press <ENTER>. LINK without the /PAUSE option performs the linking session from beginning to end without stopping.

NOTE: Do not remove the disk used for the VM.TMP file, if one has been created.

EXAMPLE: LINK file.obj/PAUSE,file.exe,,\lib\math.lib

Causes LINK to pause just before creating the executable file "file.exe." After creating the executable file, MASM pauses again to let you replace the original disk.

/SE[GMENTS]:number

The /SEGMENTS option directs LINK to process no more than the number segments per program. LINK displays an error message and stops if it encounters more than the given limit. The option is used to override the default limit of 128 segments.

The number can be any integer value in the range from 1 to 1024. It must be a decimal, octal, or hexadecimal number. Octal numbers must have a leading zero. Hexadecimal numbers must start with 0x.

If /SEGMENTS is not given, LINK allocates enough memory space to process up to 128 segments. If a program has more than 128 segments, setting the segment limit higher increases the number of segments LINK can process.

EXAMPLES: LINK file.obj/SE:10,file.exe,,;

Sets the segment limit to 10.

LINK moda+modb,run/SEGMENTS:0xff,ab.map,;

Sets the segment limit to 255 (FFH).

LINK startup+file,file/SE:030,,;

Sets the segment limit to 24 (30 octal).

/ST[ACK]:size

The /STACK option sets the program stack to the number of bytes given by size. The size can be any positive integer value in the range 1 to 65,535. The value can be a decimal, octal, or hexadecimal number. Octal numbers must begin with a zero. Hexadecimal numbers must begin with Ox.

LINK usually calculates a program's stack size automatically, basing the size on the size of any stack segments given in the object files. If /STACK is given, LINK uses the given size in place of any value it may have calculated.

LINK displays an error message if the program has no stack segments. To avoid this message, all programs should define at least one stack segment.

EXAMPLES: LINK file.obj/STACK:512,file.exe,,;

Sets the stack size to 512 bytes.

LINK moda+modb, run/ST:Oxff,ab.map,\lib\start;

Sets the stack size to 255 (FFH) bytes.

LINK startup+file/ST:030,file,,;

Sets the stack size to 24 (30 octal) bytes.

SAMPLE LINK SESSION

This sample shows you the type of information that is displayed during a LINK session.

In response to the DOS prompt, type:

```
LINK
```

The system displays a copyright message and the following prompts (responses are displayed in bold type):

```
Object Modules [.OBJ]: IO SYSINIT
Run File [IO.EXE]: /MAP
List File [C:IO.MAP]: PRN/LINE
Libraries [.LIB]: ;<ENTER>
```

Consider how the responses direct LINK and affect the output:

- By specifying /MAP, you get both an alphabetic listing and a chronological listing of public symbols.
- By responding PRN to the "List File:" prompt, you can redirect your output to the printer.
- By specifying the /LINE option, LINK gives you a listing of all line numbers for all modules. (Note that the /LINE option can generate a large volume of output.)
- By pressing <ENTER> in response to the "Libraries:" prompt, an automatic library search is performed.

Once LINK locates all libraries, the linker map displays a list of segments in the order of their appearance within the load module. The list might look like this:

START	STOP	LENGTH	NAME	CLASS
00000H	009ECH	09EDH	PROCESS	CODE
009F0H	01166H	0777H	SYSINIT	CODE

The information in the Start and Stop columns shows the 20-bit hexadecimal address of each segment relative to location zero. Location zero is the beginning of the load module.

Because the /MAP option was used, LINK displays the public symbols by name and value. For example:

```
ADDRESS    PUBLICS BY NAME

009F:0012  BUFFERS
009F:0005  CURRLOC  (Current DOS Location)
009F:0011  DEFDRV   (Default Drive)
009F:000B  DEVLIST  (Device List)
009F:0013  FILES
009F:0009  FINALLOC (Final DOS Location)
009F:000F  MEMSIZE  (Memory Size)
009F:0000  SYSINIT

ADDRESS    PUBLICS BY VALUE

009F:0000  SYSINIT
009F:0005  CURRLOC
009F:0009  FINALLOC
009F:000B  DEVLIST
009F:000F  MEMSIZE
009F:0011  DEVDRV
009F:0012  BUFFERS
009F:0013  FILES
```

The addresses displayed are not the absolute addresses where these segments are loaded. To find the absolute address of a segment, you must determine where the segment listed as being at relative zero is actually loaded. Then add the absolute address to the relative address shown in the LINK map.

Use the DEBUG utility described in Section 7 to determine where the segment listed at relative zero is actually loaded. Use these steps:

1. Load the application. Note the segment value in CS and the offset within that segment to the entry point shown in IP. (The last line of the linker map describes this entry point as relative values instead of the absolute values provided by CS and IP.)

2. Subtract the relative entry at the end of the map listing from the value in CS:IP. For example:

   CS is at 05DC
   IP is at 0

   The linker map shows the entry point as 0100:0000. The segment ID or paragraph number is 0100; the offset into that segment is 0000.

   If relative zero is located at 04DC:0000, the absolute address is 04DC0.

If a program is loaded low, the relative zero location is located at the end of the Program Segment Prefix, in the location DS plus 100H.

The beginning of the load module in addresses for public symbols is indicated by its location relative to zero:

:<offset>

If the area being referenced is relative to a segment base pointing to a segment below the relative zero beginning of the load module, the pointer becomes effectively negative. For example, the following entry appears to be the address of a load module that is almost 1 megabyte in size although it actually is not:

F8CC:EBE2H

The following message displays when LINK is completed:

Program entry at 0003:0000

LINK MESSAGES

Messages are displayed both on the monitor, and in the list file. If you direct the list file to the CON device, however, display messages are suppressed.

All messages (except for warning messages) cause the LINK session to end. (Refer to Appendix G for specific LINK messages.) After correcting a problem, you must rerun LINK.

Section 7
DEBUG UTILITY

In this section:

OVERVIEW OF DEBUG

The DEBUG utility (DEBUG) is a debugging program that provides a controlled testing environment for binary and executable object program files. DEBUG performs the following functions:

- Allows you to enter commands to load, alter, display and execute object coding in memory, and to read or write object program files.

- Facilitates the development and testing of programs coded in the MS-MACRO Assembler language.

- Provides a tool for examining memory and files to aid in resolution of system problems in abnormal situations.

For example, if you are debugging a program called FILE.EXE, the following is a typical command to start DEBUG:

```
DEBUG FILE.EXE
```

DEBUG loads FILE.EXE into memory starting at 100 hexadecimal in the lowest available segment. The BX:CX registers are loaded with the number of bytes placed into memory.

DEBUG responds with the hyphen (-) prompt to notify you to enter a command.

You can specify an <arglist> if <filename> is present. The <arglist> is a list of file name parameters and switches that are to be passed to the program <filename>. Thus, <filename> loads into memory as if it had been started with the command:

```
<filename> <arglist>
```

You can abort a DEBUG command at any time by pressing <Ctrl-C>. <Ctrl-S> pauses a scrolling display, so that you can read it. Press any key other than <Ctrl-C> or <Ctrl-S> to restart the display. All of these commands are consistent with the control character functions available at the DOS command level.

If you enter DEBUG without <filename> or arguments, you enter the DEBUG program with no specific file loaded. You can then use DEBUG commands to display ROM memory or operating system memory, or perform other functions to resolve problems.

IMPORTANT

When you start DEBUG, it sets up a program header at offset 0 in the program work area. You can overwrite the default header if no <filespec> is given to DEBUG. 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, or reenter DEBUG from DOS for it to run properly.

COMMANDS

Command Conventions

Each DEBUG command consists of a single letter followed by one or more parameters. Additionally, the control characters and the special editing functions described in Section 4 of this manual apply within DEBUG.

If a format error occurs in a DEBUG command, DEBUG reprints the command line, and indicates the error with a circumflex (^) and the word "Error."

For example:

```
DCS:100 CS:110
  ^ Error
```

You can use any combination of uppercase and lowercase letters in DEBUG commands and parameters.

Command Parameters

All DEBUG commands accept parameters, except the Quit command.

Types of DEBUG parameters are defined as follows:

Parameter	Definition
<drive>	A one-digit hexadecimal value to indicate which drive a file is loaded from or written to. The valid values are 0 to 3. These values designate the drives as follows: 0=A:, 1=B:, 2=C:, 3=D:.
<bytes>	A two-digit hexadecimal value to be placed in or read from an address or register.
<record>	A one- to three-digit hexadecimal value used to indicate the logical record number on the disk and the number of disk sectors to be written or loaded. Logical records correspond to sectors. However, their numbering differs since they represent the entire disk space.
<value>	A hexadecimal value up to four digits used to specify 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 plus an offset value. The segment designation or segment address can be omitted, in which case the default segment is used. DS is the default segment for all commands except G, L, T, U, and W, for which the default segment is CS. All numeric values are hexadecimal. For example: CS:0100 04BA:0100 The colon is required between a segment designation (whether numeric or alphabetic) and an offset.

Parameter	Definition
<range>	Two addresses: such as, <address> <address>; or one <address>, an L, and a <value>. For example: <address> L <value> where <value> is the number of lines the command should operate on, and L80 is assumed. You cannot use the last form if another hexadecimal value follows the <range>, since the hexadecimal value would be interpreted as the second <address> of the <range>. Examples: CS:100 110 CS:100 L 10 CS:100 The following is illegal: CS:100 CS:110 ^ Error The limit for <range> is 10000 hexadecimal. To specify a <value> of 10000 hexadecimal within four digits, type 0000 (or 0).
<list>	A series of <byte> values or of <string>s. <list> must be the last parameter on the command line. For example: FCS:100 42 45 52 54 41

Parameter	Definition
<string>	Any number of characters enclosed in quotation marks or single quotation marks. Within a string, you can use the opposite set of quotation marks freely. If the delimiter quotation marks must appear within a string, the quotation marks must be doubled. For example, the following strings are legal: 'This is a "string" is okay.' "This is a 'string' is okay." 'This is a ''string'' is okay.' "This is a "'string'" is okay." However, these strings are illegal: 'This is a 'string' is not.' "This is a "string" is not." Note that the quotation marks are not necessary 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.

You can separate parameters with delimiters (spaces or commas), but a delimiter is required only between two consecutive hexadecimal values. Thus, the following commands are equivalent:

```
DCS:100 110
D CS:100 110
D,CS:100,110
```

Note that not all commands use all parameters. Individual commands and their specific parameters are discussed under "Command Descriptions" in this section, where commands are arranged in alphabetical order for ease of reference.

SYSTEM REQUIREMENTS

The DEBUG utility requires:

- A memory minimum that is program dependent:

  13 KB for code
  Run space is program-dependent.

- Disk drive(s):

  One disk drive if, and only if, output is sent to the same physical disk from which the input was taken. DEBUG does not allow time to swap disks during operation on a one-diskette drive configuration. Therefore, two diskette drives or a hard disk and at least one diskette drive is a more practical configuration than a single diskette drive system.

COMMAND DESCRIPTIONS

Following are detailed descriptions of each of the DEBUG commands. Refer to Table A-5 in Appendix A for a summary of DEBUG commands.

A(SSEMBLE)

Assembles 8086/8087/8088 mnemonics directly into memory.

FORMAT: A[<address>]

COMMENTS: If a format error is found, DEBUG responds with

```
    ^ Error
```

and redisplays the current assembly address.

All numeric values are hexadecimal and must be entered as 1 to 4 characters. You must 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. These can be overridden 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
```

The NEAR prefix can be abbreviated to NE, but the FAR prefix cannot be abbreviated.

DEBUG cannot tell whether some operands refer to a word memory location or to a byte memory location. In this case, you must state the data type with the prefix WORD PTR or BYTE PTR. Acceptable abbreviations are 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.
```

Two popular pseudo-instructions are available with Assemble. 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 register indirect commands. For example:

```
ADD  BX,34[BP+2].[SI-1]
POP  [BP+DI]
PUSH [SI]
```

All opcode synonyms are also supported. For example:

```
LOOPZ  100
LOOPE  100

JA     200
JNBE   200
```

For 8087 opcodes, the WAIT or FWAIT must be explicitly specified. For example:

```
FWAIT FADD ST,ST(3)  This line assembles an
                     FWAIT prefix.

LD TBYTE PTR [BX]    This line does not.
```

C(OMPARE)

Compares the portion of memory specified by <range> to a portion of the same size beginning at <address>.

FORMAT: C<range> <address>

COMMENTS: If the two areas of memory are identical, there is no display and DEBUG returns with the DOS prompt. If there are differences, they are displayed in this format:

<address1> <byte1> <byte2> <address2>

EXAMPLE: The following commands have the same effect:

```
C100,1FF 300
```

or

```
C100 L100 300
```

Each command compares the block memory from 100 to 1FFH with the block of memory from 300 to 3FFH.

D(UMP)

Displays the contents of the specified region of memory.

FORMAT: D[<range>]

COMMENTS: If you specify a range of addresses, the contents of the range are displayed. If you type the D command without parameters, 128 bytes are displayed at the first address (DS:100) after the address is displayed by the previous Dump command.

The dump is displayed 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 denoted by a period (.) in the ASCII portion of the display. Each display line shows 16 bytes with a hyphen between the eighth and ninth bytes. At times, displays are split in this manual to fit them on the page. Each displayed line begins on a 16-byte boundary.

If you type the command:

```
DCS:100 110
```

DEBUG displays the dump in the following format:

```
04BA:0100 42 45 52 54 41 ... 4E 44 TOM SAWYER
```

If you type:

```
D
```

the display is formatted as described above. Each line of the display begins with an address, incremented by 16 from the address on the previous line. Each subsequent D (typed without parameters) displays the bytes immediately following those last displayed.

If you type:

```
DCS:100 L 20
```

the display is formatted as described on the previous page, but 20H bytes are displayed.

If you then type:

```
DCS:100 115
```

the display is formatted as described on the previous page, but all of the bytes in the range of lines from 100H to 115H in the CS segment are displayed.

E(NTER)

Enters byte values into memory at the specified <address>.

FORMAT: E<address>[<list>]

COMMENTS: If you type the optional <list> of values, the replacement of byte values occurs automatically. (If an error occurs, no byte values are changed.)

If you type the <address> without the optional <list>, DEBUG displays the address and its contents, repeats the address on the next line, and waits for your input. At this point, the Enter command waits for you to perform one of the following actions:

1. Replace a byte value with a value. Type the value after the current value. If the value typed in is not a legal hexadecimal value or if more than two digits are typed, the illegal or extra character is not echoed.

2. Press the <SPACE> bar to advance to the next byte. To change the value, type the new value as described in action 1. 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 to 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 <ENTER> key to terminate the Enter command. You can press the <ENTER> key at any byte position.

EXAMPLE: If you type 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:

```
04BA:0100  EB.41   10.  00.  BC. 42
```

Now, realizing that 10 should be 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 the <ENTER> key to end the Enter command and return to the DEBUG command level.

F(ILL)

Fills the addresses in the <range> with the values in the <list>.

FORMAT: F<range> <list>

COMMENTS: If the <range> contains more bytes than the number of values in the <list>, the <list> is repeated 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> will be ignored. If any of the memory in the <range> is not valid (bad or nonexistent), the error occurs in all succeeding locations.

EXAMPLE: If 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.

G(O)

Executes the program currently in memory.

FORMAT: G[=<address>[<address>...]]

COMMENTS: If you type only the Go command, the program executes as if the program had run outside DEBUG.

If =<address> is set, execution begins at the address specified. The equal sign (=) is required, so that DEBUG can distinguish the start =<address> from the breakpoint addresses. If this option is chosen, be certain that there is an executable instruction at the =<address>. Otherwise unpredictable results can occur.

With the other optional addresses set, execution stops at the first <address> encountered, regardless of that address position in the list of addresses to halt execution or program branching. When program execution reaches a breakpoint, the registers, flags, and decoded instruction are displayed for the last instruction executed. (The result is the same as if you had typed the Register command for the breakpoint address.)

Up to 10 breakpoints can be set. Breakpoints can be set only at addresses containing the first byte of an 8086 opcode. If more than 10 breakpoints are set, DEBUG returns the BP Error message.

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, the operating system may crash.) An interrupt code (OCCH) is placed at the specified breakpoint address(es).

When an instruction with the breakpoint code is encountered, all breakpoint addresses are restored to their original instructions. If execution is not halted at one of the breakpoints, the interrupt codes are not replaced with the original instructions.

EXAMPLE: If 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, after which the Go command is terminated.

After a breakpoint has been encountered, if you type the Go command again, the program executes just as if you had typed the file name at the DOS command level. The only difference is that program execution begins at the instruction after the breakpoint rather than at the usual start address.

H(EX)

Performs hexadecimal arithmetic on the two parameters specified.

FORMAT: H<value> <value>

COMMENTS: First, Hex adds the two parameters, then subtracts the second parameter from the first. The results of the arithmetic are displayed on one line; first the sum, then the difference.

EXAMPLE: If you type the following command:

```
H19F 10A
```

Hex performs the calculations, and displays the result:

```
02A9 0095
```

I(NPUT)

Inputs and displays one byte from the port address specified by <value>.

FORMAT: I<value>

COMMENTS: A 16-bit port address is allowed.

EXAMPLE: If you type the following command:

```
I2F8
```

If the byte at the port is 42H. Input inputs the byte, and displays the value:

```
42
```

L(OAD)

Loads a file into memory.

FORMAT: L[<address> [<drive> <record> <record>]]

COMMENTS: BX:CX are set to the number of bytes read. The file must have been named either when DEBUG was started or with the N command. Both the DEBUG invocation and the N command format a file name properly in the normal format of a file control block at CS:5C.

If you type the Load 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 Load command with an address parameter, loading begins at the memory <address> specified. If you type Load with all parameters, absolute disk sectors are loaded, not a file. The <record>s are taken from the <drive> specified (the drive designation is numeric here--0=A:, 1=B:, 2=C:, etc.); DEBUG begins loading with the first <record> specified, and continues until the number of sectors specified in the second <record> have been loaded.

EXAMPLE: Type the following commands:

```
A>DEBUG
-NFILE.COM
```

Now, to load FILE.COM, type:

```
L
```

DEBUG loads the file, and displays the DEBUG prompt.

To load only portions of a file, or certain records from a disk, type:

```
L04BA:100 2 OF 6D
```

DEBUG then loads 109 (6D hexadecimal) records from drive C: beginning with logical record number 15 into memory beginning at address 04BA:0100. When the records have been loaded, DEBUG returns the hyphen (-) prompt.

If the file has a .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 file named by the Name command or specified when DEBUG is started is a .HEX file, typing the Load command with no parameters causes DEBUG to load the file beginning at the address specified in the .HEX file. If the Load command includes the option <address>, DEBUG adds the <address> specified in the Load command to the address found in the .HEX file to determine the start address for loading the file.

M(OVE)

Moves the block of memory specified by <range> to the location beginning at the <address> specified.

FORMAT: M<range> <address>

COMMENTS: Overlapping moves (i.e., moves where part of the block overlaps some of the current addresses) are always performed without loss of data. Addresses that could be overwritten are moved first. The sequence for moves from higher addresses to lower addresses is to move the data beginning at the block's lowest address, and then to work towards the highest. The sequence for moves from lower addresses to higher addresses is to move the data beginning at the block's highest address and to work toward the lowest.

If the addresses in the block being moved will not have new data written to them, the data that is there before the move will remain. The Move command copies the data from one area into another, in the sequence described, and writes over the new addresses. This is why the sequence of the move is important.

EXAMPLE: If you type:

```
MCS:100 110 CS:500
```

DEBUG first moves 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 should type the Dump command, using the <address> typed for the Move command, to review the results of the move.

N(AME)

Sets file names.

FORMAT: N [[d:][pathname]filename[.ext]...]

COMMENTS: The Name command performs two functions. First, Name is used to assign a file name for a later Load or Write command. Thus, if you start DEBUG without naming any files to be debugged, the N<filespec> command must be typed before a file can be loaded. Second, Name is used to assign file name parameters to the file being debugged. 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
```

Because of the effects of the Name command, Name performs the following steps:

1. (N)ame assigns the file name FILE1.EXE to the file name to be used in any later Load or Write commands.

2. (N)ame also assigns the file name FILE1.EXE to the first file name parameter used by any program that is later debugged.

3. (L)oad loads FILE1.EXE into memory.

4. (G)o causes FILE1.EXE to be executed with FILE1.EXE as the single file name 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 look like this:

```
-NFILE1.EXE
-L
-NFILE2.DAT FILE3.DAT
-G
```

Here, Name sets FILE1.EXE as the file name 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 to be used by FILE1.EXE. Finally, when the Go command is executed, FILE1.EXE is executed as if FILE1.EXE FILE2.DAT FILE3.DAT had been typed at the DOS command level. Note that if you entered a Write command at this point, FILE1.EXE (the file being debugged) would be saved with the name FILE2.DAT! To avoid such undesired results, you should always execute a Name command before either a Load or a Write command.

There are four regions of memory that can be affected by the Name command:

```
CS:5C  FCB for file 1
CS:6C  FCB for file 2
CS:80  Count of characters
CS:81  All characters typed
```

A File Control Block (FCB) for the first file name parameter given to the Name command is set up at CS:5C. If you type a second file name parameter, an FCB is set up for it beginning at CS:6C. The number of characters typed in the Name command (exclusive of the first character, N) is given at location CS:80. The actual stream of characters given by the Name command (again, exclusive of the letter N) begins at CS:81. Note that this stream of characters may contain switches and delimiters that would be legal in any command typed at the DOS command level.

EXAMPLE: A typical use of the Name command is:

```
DEBUG PROG.COM
-NPARAM1 PARAM2/C
-G
-
```

In this case, the Go command executes the file in memory as if the following command line had been typed:

```
PROG PARAM1 PARAM2/C
```

Testing and debugging therefore reflect a normal run-time environment for PROG.COM.

O(UTPUT)

Sends the <byte> specified to the output port address specified by <value>.

FORMAT: O<value> <byte>

COMMENTS: A 16-bit port address is allowed.

EXAMPLE: Type:

```
O2F8 4F
```

DEBUG outputs the byte value 4F to output port 2F8.

P(ROCEED)

Causes the execution of a subroutine call, a loop instruction, an interrupt, or a repeat string instruction to stop at the next instruction.

FORMAT: P[=<address>][<value>]

COMMENTS: When at a subroutine call, a loop instruction, an interrupt, or a repeat string instruction, issue the Proceed command to execute the instruction and return control at the next instruction. Proceed command single-steps through a program, but it skips examination of the details of subroutine and interrupt calls. Each iteration of string and loop instructions is not displayed. The PROCEED command has the same syntax as the TRACE command. Specifying P0 is the same as specifying T0.

EXAMPLE: If the following instructions were executed:

```
0100  CALL    1000
0103  JC      2000
.
.
.
1000  XOR     AX,AX
.
.
.
1XXX  RET
```

and CS:IP was pointing to the CALL 1000 instruction, typing P causes the execution of the subroutine to stop and returns control to DEBUG at the JC instruction.

Q(UIT)

Q(UIT)

Terminates the DEBUG command.

FORMAT: Q

COMMENTS: The Q command takes no parameters and exits DEBUG without saving the current file. You are returned to the DOS command level.

EXAMPLE: To end the debugging session, type:

```
Q
```

```
DEBUG has been terminated, and control returns to
the DOS command level.
```

R(EGISTER)

Displays the contents of one or more CPU registers.

FORMAT: R[<register-name>]

COMMENTS: If you do not type a <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 is displayed in hexadecimal, and a colon appears as a prompt. Type either a <value> to change the register, or press the <ENTER> key if you do not want a change.

The only valid <register-name>s are:

```
AX  BP  SS
BX  SI  CS
CX  DI  IP  (IP and PC both refer
DX  DS  PC  to the Instruction Pointer.)
SP  ES  F
```

Any other entry for <register-name> results in a BR Error message.

If F is entered as the <register-name>, DEBUG displays each flag with a two-character SET/CLEAR alphabetic code (shown in the table below). To alter any flag, type the opposite two-letter SET/CLEAR code. The flags are either set or cleared.

The flags are listed below with their codes for SET and CLEAR:

Flag Name	Set	Clear
Overflow	OV	NV
Direction	DN Decrement	UP Increment
Interrupt	EI Enabled	DI Disabled
Sign	NG Negative	PL Plus
Zero	Z	NZ
Auxiliary Carry	AC	NA
Parity	PE Even	PO Odd
Carry	CY	NC

When you type the command RF, the flags are displayed in the order shown above, in a row at the beginning of a line. At the end of the list of flags, DEBUG displays a hyphen (-). You can enter new flag values at alphabetic pairs, and in any order. You do not have to leave spaces between the flag entries. To exit the R command, press the <ENTER> key. Flags for which new values were not entered 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 above, DEBUG returns a BF Error message. In both cases, the flags up to the error in the list are changed; flags at and after the error are not.

At start-up, 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.

Type:

```
R
```

R(EGISTER)

DEBUG displays all registers, flags, and the decoded instruction for the current location. If the location is CS:11A, the display appears similar to the following:

```
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<ENTER>
```

DEBUG responds only with the DEBUG prompt. To see the changes, type either the R or RF command:

```
RF
NV UP EI PL NZ AC PE CY -
```

Press <ENTER> to leave the flags this way, or enter another RF command to specify different flag values.

S(EARCH)

Searches the <range> specified for the <list> of bytes specified.

FORMAT: S<range> <list>

COMMENTS: 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.

EXAMPLE: If you type:

```
SCS:100 110 41
```

DEBUG displays a response similar to the following:

```
04BA:0104
04BA:010D
```

To search the same range of addresses for a match with the 4-byte list (41 "AB" E), type:

```
SCS:100 L 11 41 "AB" E
```

The starting addresses of all matches are listed. If no match is found, no address is displayed.

T(RACE)

Executes one instruction and displays the contents of all registers and flags, and the decoded instruction.

FORMAT: T[=<address>][<value>]

COMMENTS: If you type the optional =<address>, tracing occurs at the =<address> specified. The optional <value> causes 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 may also trace instructions stored in ROM (Read Only Memory).

EXAMPLE: Type:

```
T
```

DEBUG returns a display of the registers, flags, and decoded instruction for that one instruction. If the current position is 04BA:011A; DEBUG might return the following 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 16 (10 hex) instructions beginning at 011A in the current segment, and displays all registers and flags for each instruction as it is executed. The display scrolls away 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 point, so that you can study the registers and flags for any instruction.

U(NASSEMBLE)

Disassembles bytes and displays the source statements that correspond to them, with addresses and byte values.

FORMAT: U[<range>]

COMMENTS: The display of disassembled code looks similar to the listing for an assembled file. If you type the U command without parameters, 20 hexadecimal bytes are disassembled at the first address after that displayed by the previous UNASSEMBLE command. If you type the U command with the <range> parameter, DEBUG disassembles all bytes in the range. If the <range> is given as an <address> only, 20H bytes are disassembled instead of 80H.

EXAMPLE: 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 type

```
U04BA:0100 0108
```

The display appears as follows:

```
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 the bytes in some addresses are altered, the unassembler alters the instruction statements. You can type the U command for the changed locations, the new instructions viewed, and the unassembled code used to edit the source file.

W(RITE)

Writes the file being debugged to a disk file.

FORMAT: W[<address>[<drive> <record> <record>]]

COMMENTS: If you type Write 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 Write command with just an address, the file is written beginning at that address. If you have used a Go or Trace command, you must reset BX:CX before using the Write command without parameters. If a file is loaded and modified, 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 have been named either with the DEBUG command, or with the Name command. (Refer to the Name command description, earlier in this manual.) Both the DEBUG invocation and the Name command format a file name properly in the normal format of a file control block at CS:5C.

If you type the Write command with parameters, the write begins from the memory address specified. The file is written to the <drive> specified (the drive designation is numeric where 0=A:, 1=B:, 2=C:, etc.). DEBUG writes the file beginning at the logical record number specified by the first <record>; DEBUG continues to write the file until the number of sectors specified in the second <record> have been written.

Files having a .HEX or .EXE extension cannot be written using DEBUG Write. Therefore, if you want to make patches to these files you must first rename the files, make the patches to the renamed file and again rename the file back to its original name.

IMPORTANT

Writing to absolute sectors is EXTREMELY dangerous because the process bypasses the file handler.

Type:

```
W
```

DEBUG writes the file to disk, and displays the DEBUG prompt. Two examples are shown below.

```
W
-

WCS: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 in disk logical record number 37H and consists of 2BH records. When the write is complete, DEBUG displays the prompt:

```
WCS:100 1 37 2B
-
```

Section 8
DEFINING THE SYSTEM CONFIGURATION

In this section:

Every time you start up your PC, DOS searches the root directory of the disk from which you start for a configuration file called CONFIG.SYS. You can use this file to include information for DOS about the setup of your system.

CONFIGURATION CHANGE FEATURE

There are a number of commands that you can specify in the configuration file to configure your system for DOS. These commands perform the following functions:

- Set up more frequent checks for <Ctrl-Break> or <Ctrl-C> to stop a command or program (BREAK)
- Specify the number of disk buffers available to programs (BUFFERS)
- Specify the country whose date, time, and number conventions you want to use (COUNTRY)
- Install drivers for specific devices (DEVICE)
- Specify the number of files that can be opened by File Control Blocks (FCBS)
- Specify the maximum number of files that can be open at once (FILES)
- Set the last drive letter you can reference (LASTDRIVE)
- Specify the name of the top-level command processor (SHELL)
- Specify the number of stack resources available (STACK).

The DEVICE command defines various device drivers for DOS. DOS comes with a number of device drivers. These include:

ANSI.SYS	provides you with the ability to use extended display and keyboard control functions.
DISPLAY.SYS	supports code page switching for the monitor.
EMMDRV.SYS	supports emulation of expanded memory in extended memory.
PRINTER.SYS	supports code page switching on the parallel printer port.
DRIVER.SYS	defines the device drivers being used for external drives.
SMARTDRV.SYS	supports disk caching.
VDISK.SYS RAMDRIVE.SYS	use portions of memory to simulate diskettes for temporary storage of files and directories.

You specify these commands in the CONFIG.SYS file as options of the DEVICE command, as follows:

```
DEVICE=DRIVER.SYS
```

At some point, you may want to replace the standard DOS processor, COMMAND.COM (temporarily or permanently), with a different command processor you have developed or acquired. The SHELL command provides this capability.

This section provides an overview of the use of the configuration commands. The configuration commands are individually described in Section 3.

TABLE OF COMMANDS

Table 8-1 summarizes the commands that you can enter into a CONFIG.SYS file, and the purpose and format of each command.

Table 8-1. Configuration Commands

Name	Purpose/Format
BREAK	Allow more frequent tests for <Ctrl-Break> or <Ctrl-C> **BREAK=[ON | OFF]**
BUFFERS	Specify the number of storage buffers in memory **BUFFERS=nn**
COUNTRY	Select date, time, and currency formats **COUNTRY=ccc**
DEVICE	Define nonstandard device drivers **DEVICE=[d:][pathname]filename[.ext]**
FCBS	Specify the number of file control blocks open at the same time **FCBS=xxx,yyy**
FILES	Specify the number of files open at the same time **FILES=nnn**
LASTDRIVE	Define the number of installed disk drives **LASTDRIVE=x**
SHELL	Load an alternate command processor **SHELL=[d:][pathname]filename[.ext]**
STACKS	Allocate the number and size of system stacks for interrupt processing **STACKS=nn,sss**

THE CONFIGURATION FILE (CONFIG.SYS)

The configuration change feature of DOS uses the configuration file CONFIG.SYS to allow you to tailor your system with a minimum of effort. This special file must exist in the root directory of your boot disk. If the CONFIG.SYS does not exist, DOS uses the default values for the configuration commands.

The CONFIG.SYS file can only contain the DOS configuration commands. The configuration commands differ from other DOS system commands or batch commands in that they are not interactive commands. You cannot execute them from the DOS command line like DOS commands, or during processing, like batch commands (or DOS commands in a batch file). Instead, commands entered into a CONFIG.SYS file are used by the DOS initialization routines to modify the control information in memory. In that way, they can affect all subsequent processing without ever being executed in the normal command sense. (An exception to this is the BREAK command, which you can also use interactively to change or check the status of BREAK.)

The CONFIG.SYS file is loaded at the same time as DOS. If you make changes to the CONFIG.SYS file, they are not effective until the next time you start DOS. The start-up process for most PCs is as follows:

1. The Read-Only Memory Basic Input-Output System (ROM BIOS) chip in the computer is executed. This chip contains the power-on diagnostics routines, initialization routines, and the standard device-handling routines (the device drivers).

2. The power-on diagnostics are performed.

3. The initialization routines are performed. These include the zeroing of Dynamic Random Access Memory (DRAM), and the setting up in DRAM memory of the tables and control information needed by the operating system.

4. Part of the initialization process includes loading the DOS system routines, and executing the commands in the CONFIG.SYS file, if one exists. These commands modify the control information to reflect your configuration.

5. The final initialization routine loads the command processor, COMMAND.COM (or an alternate command processor, if the SHELL command is used). The BIOS then transfers control to the command processor.

Creating the CONFIG.SYS File

You can use the text editing capability described in Section 5 or the COPY CON: command option shown in the example below to create a CONFIG.SYS file. Your current (working) directory must be the root directory of the diskette or hard disk from which you plan to load DOS.

As an example, you may want to initially use the BUFFERS and DEVICE commands. You could create a new (or replacement) CONFIG.SYS file by making the following entries, beginning at a DOS prompt:

```
COPY CON: CONFIG.SYS
BUFFERS=10
DEVICE=ANSI.SYS
```

Complete each of the lines of the file with <ENTER>, and close the file on the last line with <F6> (or <Ctrl-Z>) and <ENTER>. This places the CONFIG.SYS file in the current directory, where the initialization routines can find it. If you include a DEVICE command, you must also make sure that DOS can locate the device driver file (in this case, ANSI.SYS). You can copy it to your boot diskette or provide a PATH command on your hard disk.

After you have created the CONFIG.SYS file, reboot the system. DOS reads the CONFIG.SYS file, and alters its control information to accommodate the commands in the file. Remember that CONFIG.SYS is read and used only when DOS starts up (when the computer is initially turned on, or when it is reset using <Ctrl-Alt-Del> or a reset button). Therefore, when you change the CONFIG.SYS file, the changes are not implemented until the next time you boot the system from the same disk.

Changing the CONFIG.SYS File

You can also change an existing CONFIG.SYS file either by text editing or by using the COPY CON: option. You can add, modify, or delete lines using the techniques described in Section 5. If you use the COPY CON: option, however, it replaces the existing CONFIG.SYS file with a new version of the file.

DEVICE DRIVER ROUTINES

Devices that interface with the PC system unit board include:

- The keyboard
- Diskette drives
- Hard disk drives
- Monochrome or color/graphics controllers
- Parallel printers, or any devices connected to serial/parallel communications controller ports (such as communications modems or serial printers).

All devices, standard or optional, have specific device routines that DOS uses to service the interfaces (transferring information and control signals to and from the devices).

The device drivers that DOS determines are needed by specific hardware, based on configuration switch settings in the PC or the presence of a response at a port, are automatically used by DOS when access to those devices is called for by your programs or DOS system commands. Some of the standard device drivers are provided with the PC, and do not need to be identified to DOS. You must specify others with DEVICE commands in the CONFIG.SYS file.

Device drivers are designed and programmed to provide common means for all programs and commands to communicate with the hardware interface (built-in port, expander board, etc.). Each one is designed for a specific hardware device. Driver routines perform testing to determine:

- Whether the device is ready to accept or send data
- Whether an error return has been transmitted by the device.

Because this level of detail testing and control is performed by driver routines, it is not required by each application program or DOS system command that uses the device.

In addition to physical devices, DOS can use logical devices (portions of a physical device that can be used as if they were separate devices). An example of a logical device is a DOS partition that shares a hard disk with at least one other DOS partition, or a portion of computer memory treated as a virtual disk.

Standard device drivers are included in the basic ROM BIOS chip installed in your PC. Device driver routines that you write or acquire for nonstandard devices must be present in the root directory of your boot disk (or locatable through a PATH command) as xxx.SYS files. (The DOS-supplied devices ANSI.SYS, DISPLAY.SYS, DRIVER.SYS, EMMDRV.SYS, PRINTER.SYS, RAMDRIVE.SYS, SMARTDRV.SYS, and VDISK.SYS were discussed at the beginning of this section.) The presence of each driver file, and your intent to have DOS use it, is defined to DOS in a DEVICE command that you enter in the CONFIG.SYS file. You can enter as many DEVICE commands as you need in your CONFIG.SYS file.

Bull HN Information Systems Inc.
Corporate Headquarters: 300 Concord Rd., Billerica, MA 01821
U.S.A.: 200 Smith Street, MS 486, Waltham, Massachusetts 02154
Mexico: Hamburgo No. 64, Col. Juarez Deleg. Cuauhtemoc, 06600 Mexico, D.F.
Asia: 4/F, Shui on Centre, 6-8 Harbour Road, Wanchai, Hong Kong
U.K.: Great West Road, Brentford, Middlesex TW8 9DH, England
Canada: 155 Gordon Baker Rd., North York, Ontario M2H 3P9
Australia: 124 Walker St., North Sydney, N.S.W. 2060
New Zealand: 14/16 Liverpool St., Auckland 1
Italy: 32 Via Pirelli, 20124 Milan

51045, 0289, Printed in U.S.A. HU95-01