MS-DOS Reference

Release 3.2

August 1986

COPYRIGHT © GRiD Systems Corporation
47211 Lakeview Blvd.
Fremont, CA 94538
(415) 656-4700

Manual Name: MS-DOS Reference: Release 3.2
Order Number: 029550-40
Issue Date: August 1986

The information in this document is subject to change without notice.

Neither GRiD Systems Corporation nor this document makes any expressed or implied warranty, including, but not limited to the implied warranties of merchantability, quality, or fitness for a particular purpose. GRiD Systems corporation makes no representation as to the accuracy or adequacy of this document. GRiD Systems Corporation has no obligation to update or keep current the information contained in this document.

Under no circumstances will GRiD Systems Corporation be liable for any loss or other damages arising out of use of this manual.

Contents

Chapter 7: MS-DOS Commands 7-1

List of Figures and Tables

About This Book

MS-DOS® is an operating system required for application programs such as Lotus 1-2-3, Multiplan, WordStar, and numerous other MS-DOS programs. This manual provides reference information on the use of MS-DOS on the GRiDCase™ computer.

Before Using This Guide

This manual is a reference tool; it is not a tutorial for learning MS-DOS. If you are already familiar with MS-DOS or another personal computer operating system, the manual should provide all the information you need to use MS-DOS successfully on the GRiDCase.

If you aren't familiar with MS-DOS or any other operating system, and you want basic training in this area, ask about MS-DOS beginner classes, written tutorials, and computer-based programs available for purchase at many computer and software stores.

Related GRiD Publications

You may need to refer to the following related publication as you work with GRiD™ MS-DOS.

□ *Using MS-DOS on the GRiDCase* A booklet for users familiar with MS-DOS for the IBM® PC or PC/XT who want to start using their new GRiDCase. It explains the differences and unique features of MS-DOS for the GRiDCase. It also provides guidelines for moving current applications to the GRiDCase, and for installing new MS-DOS applications on the GRiDCase.

Typographical Considerations

This guide follows certain typographical conventions concerning keycaps and use of the keyboard. You should be aware of these conventions as you read.

Keycaps

Keycap labels are printed in a special typeface. Thus, when you see the symbol Ctrl, you should press the key labeled Ctrl, rather than type the word "Ctrl" letter by letter.

Multi-Key Operations

Throughout this manual, you are instructed to perform operations such as "press Fn-Pause." When you see a hyphen between two keycap names, press the keys in the order in which they appear. Press the first key and, while holding it down, press the next key. Thus, when you see "press Fn-Pause," press down the Fn key and, while holding it down, press the key labeled Pause. If you see three keycap names linked with hyphens, hold down the first two while pressing the the third.

Chapter 1: Introduction

This chapter introduces GRiD MS-DOS and covers the following topics:

□ How GRiD MS-DOS is different from PC-DOS.

□ The start-up (booting) process.

□ Backing up files.

GRiD MS-DOS

GRiD MS-DOS is an operating system specifically written for use on GRiD computers; it is not the same as PC-DOS or any other type of MS-DOS system.

CAUTION: Always start your system using GRiD MS-DOS. Although PC-DOS can operate on the GRiDCase, **do not** use it for the following reasons:

□ PC-DOS is a licensed product for IBM personal computers. It is illegal to use it on other computers.

□ PC-DOS does not have facilities to control power usage, which conserves the life of a battery pack.

□ PC-DOS does not have the extended facilities of the GRiD MODE command, which supports GRiD input/output devices.

□ PC-DOS does not have the HELP command, which GRiD MS-DOS provides for on-line reference information.

□ PC-DOS does not have the PC File Transfer utilities for transferring files from an IBM PC or PC/XT to a GRiDCase.

Start-Up

The process of starting up MS-DOS is often referred to as *booting*. You start up MS-DOS in the following sequence.

1. Turn on any external devices—printer, plotter, or permanent storage devices attached to your computer.
2. Insert a diskette containing the MS-DOS system files into a disk drive. Alternately, you may have the system files in ROM or on a hard disk.
3. After waiting a few seconds, turn on the power switch to the computer.

Then the computer activates MS-DOS. (Detailed instructions for the above procedure are in the Owner's Guide for your computer.)

NOTE: MS-DOS assigns a one-character identifier to each non-ROM storage device at start-up. You use the letter to identify a device in an MS-DOS command. Refer to your computer's Owner's Guide for a list of the device letter assignments for your devices.

Note that you must prepare all storage media with the FORMAT command (described in Chapter 7) before you place files on them.

Start-Up Devices

Before start-up, the required MS-DOS system files (see below) must reside on a permanent storage device, such as Read Only Memory (ROM), a diskette, or a hard disk. The device where they reside when you turn on your computer is called the *start-up* (or *boot*) *device*.

If you have more than one storage device attached, your computer looks for MS-DOS system files in a predetermined order. You can bypass the search order and select a specific device by pressing a key on the keyboard. Refer to the Owner's Guide for your computer for specific details.

Required Start-Up Files

The following files are required on the start-up device:

□ System files, which are often called "hidden" because they do not appear in a directory listing of the device, although they actually reside on it.

□ COMMAND.COM, which interprets the commands you enter.

You place the hidden files and COMMAND.COM on the start-up device using the FORMAT command with the /S option. The SYS command only places hidden files on the start-up device. (See Chapter 7 for detailed descriptions of the commands.)

You may also need the CONFIG.SYS file for certain devices to work with your computer. (See Chapter 5 for details on CONFIG.SYS.) For example, you need the CONFIG.SYS file to create a RAM disk, to add a nonstandard storage device, to use the ANSI escape sequences, or to use the PC File Transfer utilities.

Backing Up Files

To prevent the loss of your files and avoid the reentry of data, you should establish good backup and restore procedures and routinely use them.

Backup is the practice of periodically copying data to other storage media; the copied data is then available in the event the original data is damaged or lost.

Data can be lost either as you work with it in main memory (RAM) or later, through the damage or loss of the permanent storage device where your files normally reside.

Backup Procedures

At regular intervals, make backup copies of updated files from their storage device to another storage device. For example, if you're using a hard disk, you might want to keep duplicate copies of important files on diskettes. The interval of the backup depends on how often you update the file. If you update it every day, you might want to duplicate it at the end of each workday.

Here are some guidelines for planning your backup and restore procedures.

□ Make backup copies of files that would take a long time to recreate.

□ Make backup copies each time you make changes to an important file.

□ When backing up sensitive data, consider the security of the backup device. For example, if your backup copy is on a diskette, keep it in a safe place (in a locked cabinet or at home, for example).

□ Keep a record of when you make backup copies. For example, write the backup date on the diskette label. If you lose the original file, you'll want to know the last backup date. (You can also get this information with the DIR (for *directory*) command, which displays the time a file was created or last modified.)

How To Backup Files

The most efficient way to backup multiple files is to use the BACKUP, COPY, REPLACE, or XCOPY commands in association with wildcard characters. (See Chapter 7 for a description of these commands.)

Another command, DISKCOPY, is useful when you want to backup an entire floppy diskette. (This command is also described in Chapter 7.)

A Caution About Diskettes

Among the variety of diskettes available on the market, certain types can malfunction or cause you to lose data when operating from a GRiD diskette device. To avoid this problem, use only diskettes that are double-sided and double-density; 3½-inch diskettes should be certified 135 tpi (tracks per inch); 5¼-inch diskettes should be certified 48 tpi.

Chapter 2: The File Specification

The file specification (abbreviated filespec) identifies a file in an MS-DOS command; filespec has the following format:

[d:][pathname]filename[.extension]

The brackets ([]) indicate optional parameters in a filespec. MS-DOS assumes a default value for the items in brackets if you don't specify one. The following sections describe the parameters in a filespec in detail.

Device—*d:*

The abbreviation *d:* refers to either a device letter or device name, which in turn refers to either an output device (such as a printer or a plotter), a permanent storage device (such as a diskette or a hard disk), or a modem.

If you don't specify a device, MS-DOS uses the default device indicated by the device letter that appears in the system prompt.

Device Letters

A *device letter* is a one-character identifier that MS-DOS assigns to a storage device at start-up. You use the device letter when you wish to specify a particular device in an MS-DOS command.

The letter assignments vary according to the number of devices currently attached to your computer. See the *Owner's Guide* for your computer for information on how MS-DOS assigns device letters.

MS-DOS also assigns letters to the available devices when you use GRiD Server; the available letters depend on what devices are attached to your GRiD Server unit. These letters are displayed when you sign-on to the GRiD Server.

Device Names

A *device name* refers to any device attached to your computer, including the keyboard. You use these names under the following conditions:

□ When you refer to a specific device in certain MS-DOS commands.

□ When you install or use an MS-DOS application program and the program asks you to specify the name of a particular device to perform an input or output request.

Table 2-1 shows the MS-DOS device names available on the GRiDCase.

Table 2-1. *Device Names*

Name	Device
AUX	Same as COM1.
COM1	Internal serial port or the internal modem, as the serial or modem options of the MODE command determine.
COM2	Second serial adapter port.
CON	The keyboard (for input) or display (for output).
LPT1	Normally same as PRN, unless changed by the MODE command.
LPT2	Second parallel printer, if an expansion unit is connected.
LPT3	Third parallel printer, if an expansion unit is connected.
NUL	Nonexistent file. Use NUL when the format of a command or instruction requires a filename, but you don't want to create a new file.
PRN	First parallel printer. When attached to a remote server device, you can use the SERVER command to change PRN to represent a printer at the server.

Here are some examples of device names as command parameters:

Command	Result
`COPY CON BACKUP.BAT`	Copies input entered from the keyboard to the file BACKUP.BAT. To halt copy input from the keyboard, press Ctrl-Z and RETURN in succession.
`COPY BACKUP.BAT PRN`	Prints the contents of BACKUP.BAT.
`DIR > PRN`	Prints the current directory. (See Table 6-1 in Chapter 6 for an explanation of redirecting output to a device.)

Pathname

A *pathname* specifies the path MS-DOS must follow to find a file or a directory by listing the directories that precede it. It consists of one or more directory names and a filename, if a file is specified. A pathname has the following format:

[directory] [\directory . . .] [\filename]

A backslash character (\), which is formed by pressing Fn-\, separates each name. The pathname specifies a file when a filename is the last name in the pathname; otherwise, it specifies a directory. The backslash (\) at the beginning of a pathname indicates the root directory (see The Root Directory in Chapter 3).

Filename[.extension]

Filename is the name you give a file when it is created; a valid filename contains from 1 to 8 of the following characters:

A-Z 0-9 $ & # % ’ ( ) - @ { } ~ ‘ !

NOTE: You cannot use the following reserved names as a filename: AUX, COM1, COM1STAT, COM2, COM2STAT, CON, LPT1, LPT2, LPT3, NUL, or PRN.

A file can be further identified by an *extension,* which contains up to three of the same characters that are valid for a filename; a period separates the filename and its extension.

Wildcard Characters

Certain commands allow you to replace one or more characters in the filename or extension with *wildcards*; wildcards permit you to specify multiple files in one command. The two wildcard characters you can specify are the question mark (?) and the asterisk (*).

The ? Wildcard

A question mark (?) in a filename or extension indicates that any valid character, or no character, can occupy that position. For example, the MS-DOS command

```
DEL A???.TXT
```

would delete the following files:

```
A.TXT
AT.TXT
ANY.TXT
AUNT.TXT
```

The MS-DOS command

```
DIR TEST?RUN.EXE
```

lists all directory entries on the default drive that have eight characters, begin with TEST followed by any valid character, end with the letters RUN, and have an extension of .EXE. Here are some examples of files that the above DIR command might list:

```
TEST1RUN.EXE
TEST2RUN.EXE
TEST6RUN.EXE
```

The * Wildcard

An asterisk (*) in a filename or extension indicates that any character can occupy that position or any of the remaining positions in the filename or extension. For example:

```
DIR TEST*.EXE
```

lists all directory entries on the default drive with filenames that begin with the characters TEST and have an extension of .EXE. Here are some examples of files that the above DIR command might list:

```
TEST.EXE
TEST1RUN.EXE
TEST2RUN.EXE
TEST6RUN.EXE
TESTALL.EXE
```

See Table 7-1 in Chapter 7 for other examples of commands that use wildcard characters.

CAUTION: The wildcard designation *.* refers to all files in the current directory. This can be destructive when used in MS-DOS commands. For example, the command DEL *.* deletes all files on the default drive, regardless of filename or extension.

Chapter 3: Organizing Files into Directories

As you accumulate files on a hard disk, 3½-inch diskettes, or other large storage media, it becomes increasingly difficult to locate a specific file quickly and easily. This chapter describes how to use directories to avoid this problem.

This chapter covers the following topics:

□ An overview of the structure of MS-DOS directories, the root directory, and the current directory.

□ Creating a directory.

□ Directory pathnames.

□ Entering a directory.

□ Copying files to and from directories.

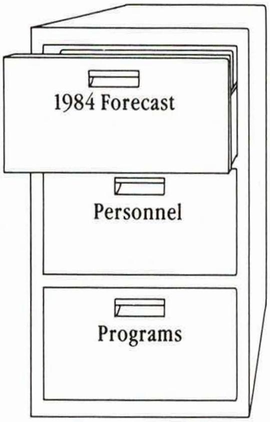
□ Exiting a directory.

□ Removing a directory.

Overview

You can organize files into logical groups with a directory. Think of each directory as a different drawer in a file cabinet that contains files organized to suit your needs. Typical directory names might include "Forecast," "Memos," "Budget," and "Employee."

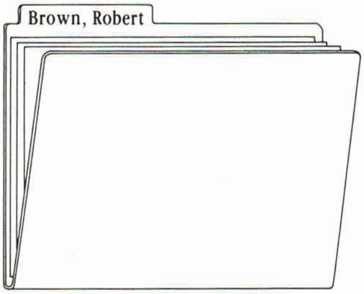
Directories Organize Files



For example, "Employee" might be a drawer, i.e., directory, in a file cabinet that contains employee file folders. The drawer can be divided by tab dividers, i.e., subdirectories, which group the folders according to status, such as current and former. Each file folder, i.e., file, is a compilation of information on individual employees, whether current or former.

Think of the filename—the name you give any collection of data—as the tab on a file folder.

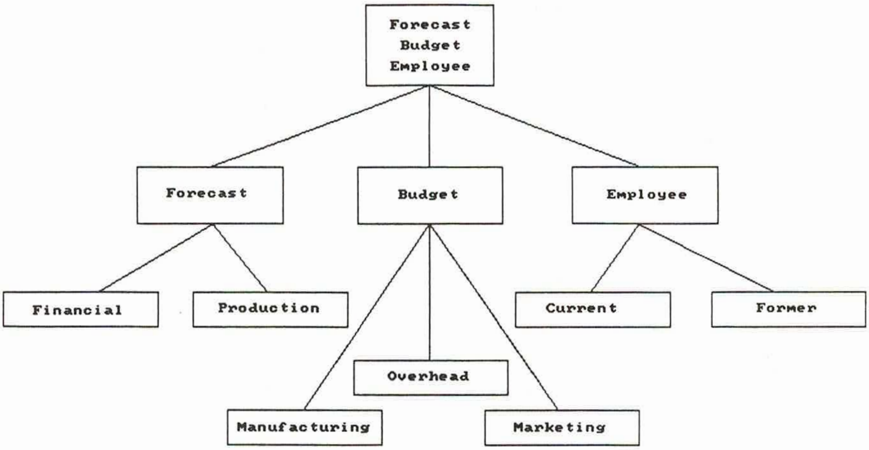
File Names Label Files



The Root Directory

MS-DOS organizes the directories you create in an inverted tree structure (Figure 3-1). At the top is the root directory, which lists all subdirectories and files in the root directory on the current MS-DOS storage device. Below the root directory are successive levels of branches, which are the successive levels of subdirectories. Directories are files that list groups of related files. You create a directory or subdirectory with the MKDIR (or MD, *make directory*) command.

Figure 3-1. *A Directory Tree Structure*



The tree structure grows as you organize groups of files into directories. Within each new directory, you can add files or new subdirectories.

MS-DOS automatically creates a root directory for each storage medium you initialize using the FORMAT command. Your hard disk and each diskette have a root directory after the formatting process.

In this guide, the term *directory* refers both to the root directory and to directories you create.

The Current Directory

You can find any file on the storage device by starting at the root directory and moving down to the directory that contains the desired file. Conversely, you can move back up to the root directory.

You enter and leave a directory using the CHDIR (or CD, *change directory*) command. The directory you enter becomes the *current directory*; if you do a DIR (*directory*) command with no parameters, MS-DOS only displays the files on the current directory.

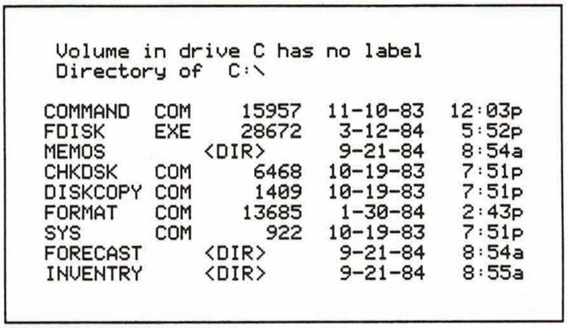
Each time you create or modify a file, or create a subdirectory, MS-DOS updates the current directory. If you haven't created any of your own directories, the current directory is the root directory.

You display the entries in a directory by entering the DIR command:

```
C>DIR
```

This command displays the entries in the current (root) directory as shown in Figure 3-2. The notation <DIR> after MEMOS, FORECAST, and INVENTRY identifies them as subdirectories.

Figure 3-2. *Directory of Files and Subdirectories*



```
  Volume in drive C has no label
  Directory of  C:\

 COMMAND  COM     15957  11-10-83   12:03p
 FDISK    EXE     28672   3-12-84    5:52p
 MEMOS        <DIR>       9-21-84    8:54a
 CHKDSK   COM      6468  10-19-83    7:51p
 DISKCOPY COM      1409  10-19-83    7:51p
 FORMAT   COM     13685   1-30-84    2:43p
 SYS      COM       922  10-19-83    7:51p
 FORECAST     <DIR>       9-21-84    8:54a
 INVENTRY     <DIR>       9-21-84    8:55a
```

Creating a Directory or Subdirectory

You create a directory or subdirectory with the MKDIR (or MD) command. For example,

```
C>MD MEMOS
```

creates a directory named *memos,* subordinate to the current directory.

NOTE: You can also create a tree of directories, each subordinate to the previous one, cascading down from *memos.* However, such complex directory structures increase MS-DOS processing time and can complicate locating a file.

Entering a Directory

You normally enter a directory when you want to use the files and applications it contains, and ignore all others on the same device. Use the CHDIR (or CD) command to enter a directory. For example,

```
C>CD MEMOS
```

makes MEMOS the current directory. You can also use this command to find out the name of the current directory. For example,

```
C>CD
```

displays the following information:

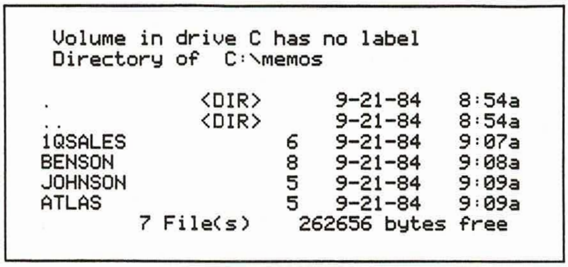
```
\MEMOS
```

where \ signifies the root directory, and MEMOS the current directory.

The files in MEMOS now appear to be the only ones on the default device. For example, if you enter the DIRECTORY command

```
C>DIR
```

MS-DOS displays only the files under MEMOS:



```
 Volume in drive C has no label
 Directory of  C:\memos

.            <DIR>      9-21-84   8:54a
..           <DIR>      9-21-84   8:54a
1QSALES             6   9-21-84   9:07a
BENSON              8   9-21-84   9:08a
JOHNSON             5   9-21-84   9:09a
ATLAS               5   9-21-84   9:09a
        7 File(s)    262656 bytes free
```

Directory Pathnames

A *pathname* consists of one or more directory names, each separated by backslash (\), and specifies either a directory or a file. The backslash joins the names of the directories and subdirectories that contain the desired directory or file. (You make a backslash by pressing Fn-\.)

The first backslash specifies the root directory; each name thereafter specifies a directory subordinate to the previous one. The last name specifies the desired directory or file. For example, the following pathname specifies the newly created MEMOS directory:

```
\MEMOS
```

The following pathname refers to the file 1QSALES in the directory MEMOS:

```
\MEMOS\1QSALES
```

You must specify a pathname in MS-DOS commands for files not in your current directory, as shown in the next section.

Copying Files to and from Directories

If you want to copy files into a directory, make that directory the current directory using the CHDIR (or CD) command. Once it is the current directory, you don't need to specify a pathname in the COPY command. For example, when the current directory is MEMOS, the following command

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

copies all files in MEMOS to the current directory of the diskette in drive A. The command

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

copies all files on the diskette in drive A into MEMOS.

However, to copy a file to or from another directory, you must specify a pathname for the second directory. For example, assuming MEMOS is the current directory, the command

```
C>COPY *.* \INVENTRY
```

copies all files in MEMOS to the directory INVENTRY. The command

```
C>COPY \INVENTRY\*.*
```

copies all files in INVENTRY to MEMOS.

NOTE: Press Fn-\ to make the backslash (\) character.

Exiting a Directory

Use the CHDIR (or CD) command to exit the current directory and go to another directory. For example, assuming that the current directory on the default device (e.g., C:) is MEMOS, the command

```
C>CD \
```

makes the root directory the current directory.

CD followed by two periods (. .) returns you to the next highest directory, the *parent directory*. Because this example uses only two levels of directories, the parent directory for memos is the root directory. Therefore, the command

```
C>CD ..
```

also makes the root directory the current directory.

The command

```
C>CD \INVENTRY
```

makes INVENTRY the current directory. See the CHDIR command in Chapter 7 for examples of moving about in multi-level directories.

Removing a Directory

Use the RMDIR (or RD, *remove directory*) to delete a subdirectory. Before you can remove a directory, you must delete all files in it. For example, if you want to remove MALE from the following directory structure

```
\EMPLOYEE\FORMER\MALE
```

make MALE the current directory. Display all the files in MALE with the DIR command. Copy all files in MALE to a backup device; then delete all files in MALE with the command DEL *.*. Use the CHDIR (or CD) command to go to the parent directory, FORMER. Remove MALE with the RMDIR (or RD) command`as follows:

```
RD MALE
```

Chapter 4: Hard Disk Preparation

This chapter explains how to prepare a hard disk for use with your computer.

Overview

To prepare a hard disk for use, do the following:

- □ Define an area on the hard disk for your system and application files. This area is called a *partition*. Use the FDISK program described in this chapter to create this partition.
- □ Format the partition and copy in your MS-DOS system and application files. Use the FORMAT command and the COPY or RESTORE command (described in Chapter 7) to carry out these functions; they work the same for a hard disk as they do for diskettes.

MS-DOS assigns a one-character identifier to each hard disk that is attached to your computer at start-up. You use this letter to identify the disk in an MS-DOS command. MS-DOS normally assigns the letter *C* to a hard disk, although the letter varies according to the number of external storage devices that are attached and the settings of their address switches. Refer to the Owner's Guide for your computer for information on the device letter assignments.

MS-DOS and GRiD-OS Partitions

GRiD Systems provides a set of utilities that install GRiD-OS (GRiD's proprietary operating system) in a partition on the same hard disk as MS-DOS. This allows you to switch from one operating system to the other with ease. GRiD-OS supports a variety of management tools, including a word processor, a spreadsheet, and graphing, database, and communications facilities.

The optional GRiD-OS/MS-DOS utilities, which operate under GRiD-OS, consist of the following programs:

- □ MediaPartition, which creates a GRiD-OS partition on the hard disk, performs many of the same functions for GRiD-OS as FDISK does for MS-DOS.
- □ GRiDTransfer transfers files between the GRiD-OS and MS-DOS operating environments. The files can reside either on diskette or in either partition of the hard disk.

These utilities are described in the *GRiD-OS/MS-DOS Utility Programs User's Guide* that comes with GRiD software. Contact your GRiD representative for information about the utilities and GRiD Management Tools.

NOTE: If you intend to install both GRiD-OS and MS-DOS partitions on your hard disk, you should run MediaPartition and install the GRiD-OS partition first.

NOTE: If you have a disk that is entirely formatted for GRiD-OS and you want to reformat it for MS-DOS, you must first run MediaPartition, create a dummy GRiD-OS partition and then delete the partition. After this is done, you can start up MS-DOS and run FDISK to format you hard disk.

The FDISK Program

To start up the FDISK program, do the following:

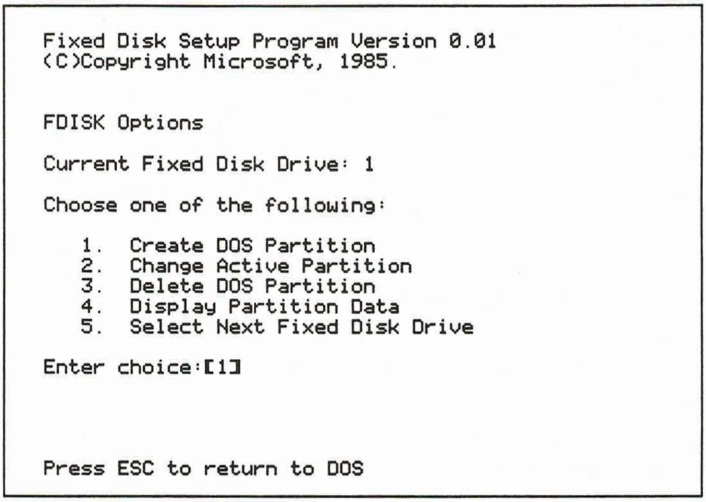
□ Enter FDISK as follows:

```
A>FDISK
```

and press Return.

□ The FDISK menu (Figure 4-1) appears.

Figure 4-1. *The Fixed Disk Options Menu*



```
Fixed Disk Setup Program Version 0.01
(C)Copyright Microsoft, 1985.

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 Data
    5.  Select Next Fixed Disk Drive

Enter choice:[1]



Press ESC to return to DOS
```

FDISK provides several functions in addition to partitioning the disk, as Figure 4-1 indicates. To select an item from the menu, enter the item number at the prompt and press Return. If you press Return without entering a number, you select the default, Option 1 (Create a DOS Partition).

Note that item 5 only appears on the menu if you have more than one hard disk attached to your computer.

To exit from the FDISK program, do the following:

- □ Press Esc once if your screen displays the FDISK Options menu.
- □ Press Esc twice if your screen displays one of the function screens listed in Figure 4-1. The first Esc returns you to the FDISK Options menu; the second Esc returns you to MS-DOS.

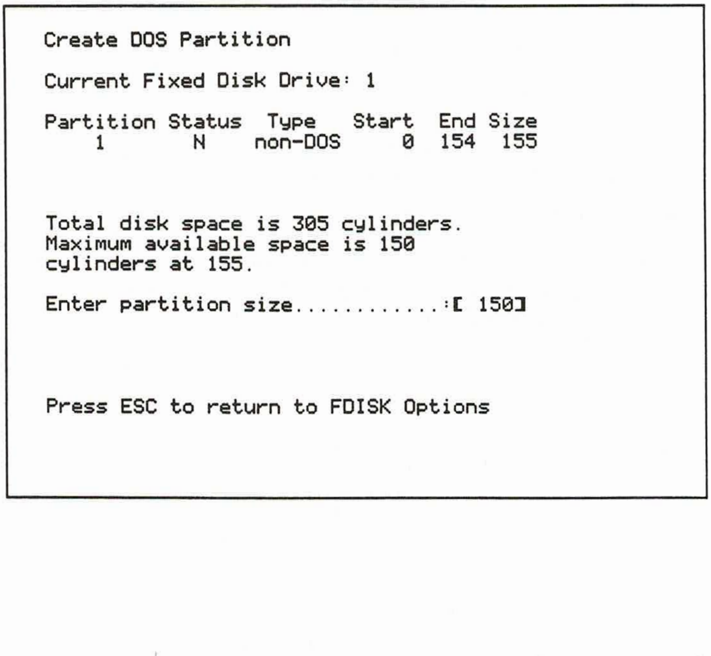
The options in the menu (Figure 4-1) are explained in the sections that follow.

CAUTION: FDISK permits the deletion and reassignment of partitions. When you delete a partition and assign a new one in its place, the files in the area allocated are lost. Therefore, back up all files from a partition area onto a diskette or other external storage device before you delete the partition.

Create a DOS Partition (Option 1)

Create a DOS Partition lets you allocate space on the disk for MS-DOS. When you select this option and press Return, the Create Screen (Figure 4-2) appears on your display.

Figure 4-2. *The Create Screen*



```
Create DOS Partition

Current Fixed Disk Drive: 1

Partition Status  Type   Start  End Size
    1       N    non-DOS     0  154  155



Total disk space is 305 cylinders.
Maximum available space is 150
cylinders at 155.

Enter partition size............:[ 150]




Press ESC to return to FDISK Options
```

The first unassigned partition in the screen becomes your partition (Partition 2 in Figure 4-2). See Display Partition Information below for a detailed explanation of the other items on the form.

If no partition is currently assigned on the hard disk, the following message appears:

Do you wish to use the entire fixed disk for DOS? (Y/N)

If you enter Y (for *yes*), the entire fixed disk is assigned to DOS.

If you enter N (for *no*), or if a partition already exists, the prompts described below are displayed. To assign the partition, respond to these prompts and press Return. (To cancel your response to the prompts, press Esc instead of Return.)

1. Enter partition size:

   The initial setting is the number of contiguous cylinders containing the largest block of free space on the hard disk. Press Return to use the initial value, or specify the number of cylinders you wish to allocate to the new partition.

   Consider the following when you specify a partition size:

   □ Do you have adequate space in the partition to handle all future MS-DOS needs? Once your drive is partitioned, it cannot be repartitioned without the time-consuming process of backing up all files.

   □ If you intend to share the hard disk with another operating system, make sure there is adequate space on the hard disk for the other operating system.

   NOTE: The GRiD Model 2201 hard disk has 305 cylinders numbered 0 through 304; each cylinder contains approximately 34,816 bytes (34K). Consult your disk drive owner's manual if you are using a different model.

   CAUTION: If you create a partition that is less than 128K bytes, it will not be recognized by the system.

2. Enter starting cylinder number:

   The initial setting is the first cylinder number where the largest block of free space begins. To use this value press Return, or else specify any appropriate cylinder number and press Return.

The new partition is created and the following messages are displayed:

```
System will now restart
Press any key when ready
Then select the boot device
```

You can restart your system from any device that contains the MS-DOS system files except the hard disk that you just partitioned. **Remember to press the appropriate key to indicate the start-up device.** For example, to start-up from the floppy drive, press the *F* key. If you have MS-DOS in ROM, you can press any other key, and the system will start-up correctly.

You must format the new partition, using the FORMAT command, before you can use it.

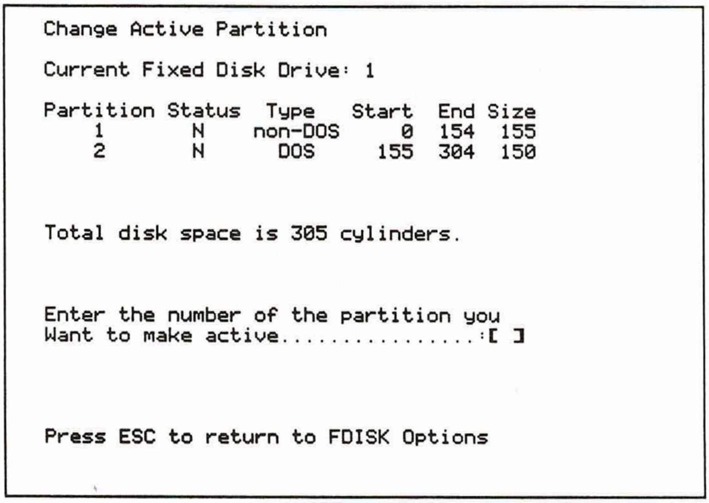
CAUTION: Attempting to use a new partition before formatting it could have unpredictable results.

Change Active Partition (Option 2)

Change Active Partition lets you instruct the computer to use a specific operating system when the hard disk is the start-up device. You designate the partition containing this operating system as the *active partition.* (The hard disk becomes the start-up device when you start up (boot) your system while holding down the H (for ***hard disk***) key on the keyboard.)

When you select Change Active Partition and press Return, the Change Active Partition screen (Figure 4-3) is displayed.

Figure 4-3. *The Change Active Partition Screen*



```
Change Active Partition

Current Fixed Disk Drive: 1

Partition Status  Type    Start  End Size
    1       N    non-DOS     0   154  155
    2       N      DOS     155   304  150



Total disk space is 305 cylinders.



Enter the number of the partition you
Want to make active................:[ ]




Press ESC to return to FDISK Options
```

Enter the number (1, 2, 3, or 4) of the active partition. You must exit normally from FDISK for the change to take effect. Note that when you exit FDISK, the hard disk becomes the active drive.

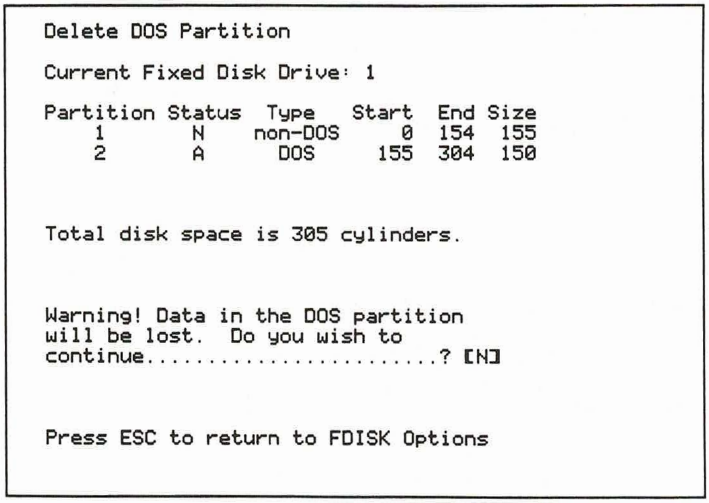
When you start up from the hard disk, the operating system is loaded from the partition you specified. This allows you to boot directly into either GRiD-OS or MS-DOS when both operating systems reside on the disk.

NOTE: If you have at least one GRiD-OS partition on your hard disk, and you have made one of your hard disk partitions active and would like to have neither of them active, you can use the GRiD MediaPartition utility. This utility includes an option that allows you to set no partitions active. When no partitions are active and you start up from the hard disk, you are prompted to select the partition you want to boot from.

Delete DOS Partition (Option 3)

The Delete Partition screen (Figure 4-4) appears when you select this option.

Figure 4-4. *The Delete Partition Screen*



```
Delete DOS Partition

Current Fixed Disk Drive: 1

Partition Status  Type    Start  End Size
    1       N    non-DOS      0  154  155
    2       A      DOS      155  304  150


Total disk space is 305 cylinders.


Warning! Data in the DOS partition
will be lost.  Do you wish to
continue........................? [N]


Press ESC to return to FDISK Options
```

The warning message shown in the figure appears, and the following prompt appears:

```
Do you wish to continue? [N]
```

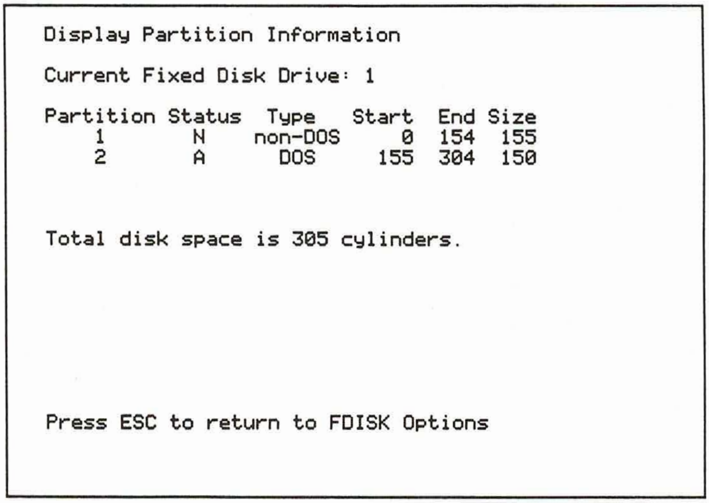
If you press Return, the command is canceled and the FDISK menu appears. If you enter Y and then press Return, the partition is deleted.

CAUTION: Any files in the area allocated to a partition are lost after you delete it. Therefore, back up these files onto a diskette or other external storage device before deleting a partition.

Display Partition Data (Option 4)

Display Partition Data displays a table (Figure 4-5) describing the partitions on a hard disk.

Figure 4-5. *Partition Information Display*



```
Display Partition Information

Current Fixed Disk Drive: 1

Partition Status  Type    Start  End Size
    1       N    non-DOS     0   154  155
    2       A      DOS     155   304  150


Total disk space is 305 cylinders.





Press ESC to return to FDISK Options
```

The items in the table are explained below.

- Partition: A number, from 1 to 4, that identifies the partition.
- Status: *A* or *N*. *A* indicates an active partition; *N* indicates the partition is not active. See the section Change Active Partition in this chapter for the definition of an active partition.
- Type: The type can be either the name of the operating system currently residing in the partition, or the word *non-DOS*, indicating that FDISK can't identify the operating system in the partition. This may indicate another operating system such as GRiD-OS.
- Start: The cylinder number of the start of the partition's data storage area.
- End: The cylinder number representing the end of the partition's data storage area.
- Size: The number of cylinders used by the partition; one cylinder occupies 34,816 bytes (34K).

Select Next Fixed Disk Drive (Option 5)

This option only appears on the FDISK Options menu if more than one hard disk is attached to your computer. Use this option to switch from one hard disk to another.

After you select this option, FDISK automatically switches to the second attached hard disk drive. To switch back to the first hard disk, select this option again.

The number of the current hard disk drive depends on how you set the address switches on your hard disks. Refer to the Owner's Guide for your computer for specific switch setting information.

Chapter 5: The CONFIG.SYS File

This chapter describes the CONFIG.SYS file and the commands available for it in GRiD MS-DOS; it covers the following topics:

□ Purpose of the CONFIG.SYS file

□ How to create a CONFIG.SYS file

□ Available commands for the CONFIG.SYS file

You must create or modify a CONFIG.SYS file under the following conditions:

□ When you wish to use the PC File transfer utilities (see Chapter 12).

□ When you wish MS-DOS to recognize special ANSI characters.

□ When you wish to use a RAM disk (extra logical disk drive using main memory).

□ When you wish to assign an additional logical drive letter to an existing attached disk drive, or install a nonstandard external device.

Purpose of CONFIG.SYS

The CONFIG.SYS file modifies certain assumptions or defaults built into MS-DOS. The file consists of configuration commands that you insert; MS-DOS subsequently inspects them during start-up and takes the requested action.

Creating a CONFIG.SYS File

You can create a CONFIG.SYS file using a text editor (such as EDLIN) just as you would any other text file. You can also create a file using the COPY command, as shown in the next procedure.

To Create a CONFIG.SYS File

1. If you have an existing CONFIG.SYS file, display the commands in the file by entering the TYPE command and pressing Return, as shown below.

```
A>TYPE CONFIG.SYS
```

(You must reenter those commands displayed, as shown in the next steps, if you wish them to remain in effect.)

2. Enter the COPY command as shown below and press Return.

```
A>COPY CON CONFIG.SYS
```

This command makes MS-DOS place all characters you enter from the keyboard into the CONFIG.SYS file.

3. Enter the commands you wish to include in the file and press Return after each one.
4. If you want to reenter commands that were in the earlier file, then enter those commands and press Return after each command.
5. Press Ctrl-Z and then Return.

   Pressing Ctrl-Z halts the insertion of characters into the CONFIG.SYS file and returns normal keyboard functions.

After you complete the above steps, you can display the CONFIG.SYS file using the TYPE command to check the accuracy of your entries.

NOTE: You must now restart your system for the new commands in your file to take effect.

The commands you can use in the CONFIG.SYS file are summarized in the next section and described in detail in the following sections.

CONFIG.SYS Command Summary Table 5-1 summarizes the commands that are available for use in the CONFIG.SYS file.

Table 5-1. *CONFIG.SYS Commands*

Command	Description
BREAK	Extends the Ctrl-C check to additional system operations.
BUFFERS	Sets the number of disk buffers allocated in memory at start-up.
COUNTRY	Sets the system to use international time, date, and currency formats.
DEVICE	Installs device drivers into the system. The standard device drivers supplied with GRiD MS-DOS are listed below.
	PCMASTER.SYS which allows use of the PC File Transfer utilities (see Chapter 11 for more details).
	ANSI.SYS which allows use of ANSI escape code sequences for output.
	RAMDRIVE.SYS which creates a RAM disk using main memory.
	DRIVER.SYS which allows you to create an extra logical drive or install a non-standard external storage device.
DRIVPARM	Defines parameters for block devices.
FCBS	Specifies the number of File Control Blocks (FCBs) that can be concurrently open.
FILES	Sets the number of open files that can access certain MS-DOS system calls.
LASTDRIVE	Sets the maximum number of drives that you may access.
SHELL	Begins execution of the shell from a specific file.

These commands are described in detail in the following sections.

BREAK

The BREAK command extends the Ctrl-C check function to additional system operations.

BREAK has the following format:

```
BREAK = ON
or
BREAK = OFF
```

Depending on the program you are running, Ctrl-C may be used to stop an activity (for example, to stop sorting a file). Normally, MS-DOS checks to see if Ctrl-C has been typed while it is reading from the keyboard, writing to the screen, or to a printer. Setting BREAK to ON allows Ctrl-C checking to be extended to other functions such as disk reads and writes. The default is BREAK = OFF.

BUFFERS

The BUFFERS command allows you to set the number of disk buffers that MS-DOS allocates in main memory (RAM) at system start-up.

BUFFERS has the following format:

```
BUFFERS = n
```

where *n* specifies the number of buffers to allocate.

A disk buffer is a block of memory where MS-DOS can hold data being read from or written to a disk when the amount of data is not an exact multiple of the sector size. For applications that frequently access the disk(s), increasing the number of buffers will result in faster performance.

The default number of buffers is 2. For applications such as word processors, a number between 10 and 20 will provide the best performance. If you plan to create many subdirectories, increase the number of buffers to a value between 20 and 30.

Note that each buffer allocated takes approximately 512 bytes of main memory. Therefore, allocating a large number of buffers decreases your available working memory.

COUNTRY

The COUNTRY command allows MS-DOS to use international time, date, currency, and case conversion.

COUNTRY has the following format:

```
COUNTRY = nnn
```

where *nnn* can have a value shown in the following table.

Value	Country
001	United States
031	Netherlands
032	Belgium
033	France
034	Spain
039	Italy
041	Switzerland
044	United Kingdom
045	Denmark
046	Sweden
047	Norway
049	Germany
061	Australia
358	Finland
972	Israel

The default country value is 001 (United States). To set the time, date, currency, and case formats to those of a different country, specify the appropriate value. For example, the command:

```
COUNTRY = 033
```

causes the time, date, currency, and case formats to use the French standards.

DEVICE

The DEVICE command specifies the name of a file containing a *device driver*. A device driver is a software component within MS-DOS that controls input/output operations to printers, plotters, diskettes, hard disks, and other storage devices. MS-DOS and MS-DOS application programs supply device drivers that support most input/output operations. You may need the DEVICE command to support MS-DOS input/output operations that are not standard.

DEVICE has the following format:

```
DEVICE = filespec
```

where *filespec* specifies a device driver file. The device drivers supplied with GRiD MS-DOS are described in the following subsections.

PCMASTER.SYS

The PCMASTER.SYS device driver allows you to use the PC File Transfer utilities to transfer files between an IBM PC, PC/XT, or PC/AT and a GRiDCase. Specify this device driver as follows:

```
DEVICE = PCMASTER.SYS [1 | 2]
```

where the *1* or *2* refers to the serial port (COM1 or COM2) you are using at the master station. See Chapter 11 for details on using the PC File Transfer utilities.

ANSI.SYS

The ANSI.SYS device driver must be installed when executing an application that uses ANSI escape code sequences. The ANSI escape code sequences are described in the Appendix. Specify this device driver as follows:

```
DEVICE = ANSI.SYS
```

RAMDRIVE.SYS

The RAMDRIVE.SYS device driver lets you use a portion of your computer's main memory (RAM) as if it were a disk drive. This memory area is called a RAM disk and is sometimes referred to as a virtual disk. RAM disks are much faster than disk drives because the information they contain is always loaded into RAM.

NOTE: When you reset or turn off your computer, the information stored in RAM disks is lost. Be sure to copy any important RAM disk files to a permanent storage device before turning off or reseting your computer.

Specify this device driver as follows:

```
DEVICE = RAMDRIVE.SYS [bbbb] [ssss] [dddd] [/E | /A]
```

The optional parameters are described below:

bbbb — Specifies the RAM disk size in kilobytes. The default value is 64, the minimum value is 16.

ssss — Specifies the sector size in bytes. The default is 128. Valid values are: 128, 256, and 512.

dddd — Specifies the number of root directory entries. The default value is 64. Valid values range from 2 to 1024.

RAMDRIVE.SYS adjusts the value of *dddd* to the nearest sector boundary. For example, if you give a value of 25 when the sector size is 512 bytes, the 25 will be rounded up to 32. This is because 32 is the next multiple of 16 (there are sixteen 32-byte directory entries in 512 bytes).

/E — Allows you to use extended memory (above 1MB) as a RAM disk if it has been installed in an expansion unit. If you use this switch, you cannot use the /A switch.

/A — Allows you to use an extended memory board that meets the Lotus®/Intel®/Microsoft® Expanded Memory Specification for a RAM drive, if such a board has been installed in an expansion unit. If you use this switch, you cannot use the /E switch.

DRIVER.SYS

The DRIVER.SYS device driver allows you to create an extra logical drive or install a non-standard external storage device. Specify this device driver as follows:

```
DEVICE = DRIVER.SYS /D:dd [/C] [/F:ff] [/H:hh] [/N] [/S:sss] [/T:ttt]
```

The parameters are described below:

/D:dd Specifies a physical drive number between 0 and 255. Physical drives are numbered differently than logical drives. Floppy disk drives are numbered starting at 0, where drive A = 0, drive B = 1, drive C = 2, etc. Hard disk drives are numbered starting at 80H, where the first drive = 80, the second drive = 81, etc.

/C Specifies that changeline (doorlock) support is required.

/F:ff Specifies the type of device attached, where *ff* is one of the following:

00 = 320/360 KB disk drive
01 = 1.2 MB disk drive
02 = 720 KB disk drive
03 = 8-inch single density disk drive
04 = 8-inch double density disk drive
05 = hard disk
06 = tape drive
07 = other

If you do not specify this switch, the default value is 02 (720 KB disk drive).

/H:hh Specifies the number of heads on the external device, from 1 to 99.

/N Specifies a non-removable block device such as a tape drive.

/S:ss Specifies the number of sectors per track, from 1 to 99.

/T:ttt Specifies the number of tracks per side on a block device, from 1 to 999.

You can use DRIVER.SYS to create an extra logical device letter for a device already attached to your computer. For example, to create another logical device letter for your first hard disk, specify the following command:

```
DEVICE=DRIVER.SYS /D:80 /F:05
```

The /D:80 is the physical drive number of the first hard disk, and /F:05 specifies the type of device (hard disk).

DRIVPARM The DRIVPARM command allows you to define parameters for block devices, overriding the original MS-DOS device driver settings.

DRIVPARM has the following format:

```
DRIVPARM = /D:dd [/C] [/F:ff] [/H:hh] [/N] [/S:ss] [/T:ttt]
```

The parameters are described below:

/D:dd Specifies a physical drive number between 0 and 255. Physical drives are numbered differently than logical drives. Floppy disk drives are numbered starting at 0, where drive A = 0, drive B = 1, drive C = 2, etc. Hard disk drives are numbered starting at 80H, where the first drive = 80, the second drive = 81, etc.

/C Specifies that changeline (doorlock) support is required.

/F:ff Specifies the type of device attached, where *ff* is one of the following:

00 = 320/360 KB disk drive
01 = 1.2 MB disk drive
02 = 720 KB disk drive
03 = 8-inch single density disk drive
04 = 8-inch double density disk drive
05 = hard disk
06 = tape drive
07 = other

If you do not specify this switch, the default value is 02 (720 KB disk drive).

/H:hh Specifies the number of heads on the external device, from 1 to 99.

/N Specifies a non-removable block device such as a tape drive.

/S:ss Specifies the number of sectors per track, from 1 to 99.

/T:ttt Specifies the number of tracks per side on the block device, from 1 to 999.

For example, you might have an external tape drive configured by the DEVICE = DRIVER.SYS command as drive D that writes 20 tracks of 40 sectors per track. If you want to reconfigure this tape drive to write 10 tracks of 99 sectors each, you can specify DRIVPARM as follows:

```
DRIVPARM=/D:3 /F:6 /H:1 /S:99 /T:10
```

This overrides the default device driver settings, and supports a tape drive as drive D. This tape drive has 1 head, and supports a tape format of 10 tracks and 99 sectors per track. You might want to use this method to create a tape that you can read on another computer that can only read this alternate format.

FCBS The FCBS command allows you to specify the number of FCBs (File Control Blocks) that can be concurrently open.

FCBS has the following format:

```
FCBS = x,y
```

where *x* represents the number of files opened by FCBs that can be open at any one time. The default value for *x* is 4. Valid values range from 1 to 255.

y represents the number of files opened by FCBs that MS-DOS cannot close automatically if an application tries to open more than *x* files by FCBs. The first *y* files opened by FCBs are protected from being closed. The default value of *y* is 0. Valid values range from 1 to 255.

Note that *y* cannot exceed *x*.

FILES

The FILES command sets the number of open file handles compatible with XENIX® that the MS-DOS system call can access.

FILES has the following format:

```
FILES = x
```

where *x* represent the number of open file handles that the system calls can access. *x* can be any number between 8 and 20. The default is 8.

System calls 2FH through 60H are compatible with the XENIX operating system.

LASTDRIVE The LASTDRIVE command sets the maximum number of drives that you may access.

LASTDRIVE has the following format:

```
LASTDRIVE = x
```

where *x* can be any letter from A to Z.

This value represents the last valid drive that MS-DOS will accept. The default value is E. The minimum number is equal to the number of drives you have installed on your computer.

This command is only useful in a network environment. At start-up, MS-DOS recognizes five drive letters: A-E, no matter how many physical drives you have on your system. A network redirection must occur to make any of the extra drives defined by LASTDRIVE valid.

Note that MS-DOS allocates a data structure for each drive that you specify, so you shouldn't specify more drives than necessary.

SHELL

The SHELL command begins execution of the shell (top-level command processor) from the specified file.

SHELL has the following format:

```
SHELL = filespec
```

where *filespec* specifies the file containing the shell.

This command can be used by system programmers who write their own command processor (the MS-DOS file named COMMAND.COM). MS-DOS will start the processor specified by *filespec* instead of reading the standard COMMAND.COM.

Chapter 6: Input and Output

This chapter describes how to control the flow of data that a command or program needs or produces. The chapter covers the following topics:

- □ Controlling the flow of input and output.
- □ Halting output to the display.
- □ Printing command output.
- □ Halting execution of a command.
- □ Command filters, which accept output from another command, process it, and send the output to a file or device you specify.
- □ Command piping—giving the system more than one command at a time.

Changing Input/Output Flow

MS-DOS assumes that input comes from the keyboard and output goes to the screen. However, you can change this flow by specifying the special characters shown in Table 6-1 in a command.

Table 6-1. *Special Characters for Changing MS-DOS Input/Output*

Character	Format	Result
>	command>filespec	Places the output of *command* in the file specified by *filespec*.
>	command>device	Directs the output of *command* to *device*, which can be any valid device name shown in Table 2-1. For example, to send output to the printer, specify PRN; to send it to the serial port, specify COM1.
>>	command>>filespec	Takes the output produced by *command* and appends it to the data in the file specified by *filespec*.
<	command<filespec	Uses the data in the file specified by *filespec* as input.

NOTE: Using the CTTY command, you can change the input device where you enter commands, which is normally the keyboard. See a description of the CTTY command in Chapter 7.

As Table 6-1 indicates, you can specify that input come from a file rather than the keyboard, and output can go to a file, printer, or other output device. The following examples show how input or output might be redirected.

Example 1: Redirecting Output

The following command displays a directory listing of the disk in the default drive:

```
A>DIR
```

You can use the same command to send this output to a file named MYFILES by designating it as the output file:

```
A>DIR>MYFILES
```

If MYFILES already exists, MS-DOS writes over it with the new data; otherwise, MS-DOS creates it and stores your directory listing in it.

If you specify

```
A>DIR>B:MYFILES
```

the output is directed to the file MYFILES on drive B.

Example 2: Appending Output

You can append your directory listing to an existing file named MYFILES with the following command:

```
A>DIR>>MYFILES
```

If MYFILES does not exist, MS-DOS creates it.

Example 3: Changing Input Source

Often it is useful to have input for a command come from a file rather than from the keyboard. In the following example, the command filter SORT sorts the file NAMES and sends the sorted output to a file named LIST1. See Command Filters below for more details.

```
A>SORT<NAMES>LIST1
```

Halting MS-DOS Processing

Pressing Fn-Pause (or Ctrl-S) temporarily halts MS-DOS display processing; then pressing any key causes MS-DOS to resume display processing.

This key is useful when displaying several pages of data on the screen; as the screen becomes full, MS-DOS scrolls the data upward to make room for each new line at the bottom of the screen. To temporarily halt scrolling, press Fn-Pause (or Ctrl-S); to resume scrolling, press any key.

For example, if you want to display the contents of a file using the TYPE command, enter the command and the name of the file you want to display:

```
A>TYPE 1QSALES
```

MS-DOS then scrolls the data in 1QSALES onto the screen; you can temporarily halt scrolling by pressing Fn-Pause (or Ctrl-S); you can resume scrolling by pressing any key.

You can achieve the same result by entering the same command, adding the MORE command filter, as follows:

```
A>TYPE 1QSALES | MORE
```

(Press Fn-Shift-| to enter the | character.)

With the MORE command filter, MS-DOS displays the data in 1QSALES one screen at a time and places the following message at the bottom of each screen display:

```
—MORE—
```

which indicates that more output remains. Press any key to display the next screen.

Printing Command Output

You may want to print, instead of display, the output of a command; you can print command output in either of the following two ways:

Method 1: Enter the command followed by the greater-than symbol (>) and the characters *PRN*; for example, the following command prints a file directory:

```
A>DIR>PRN
```

The following command prints the contents of 1QSALES:

```
A>TYPE 1QSALES>PRN
```

Method 2: Press Fn-Echo (or Ctrl-P), enter the command (*without* the PRN item), and press Return. Any output from the command is printed and displayed on the screen. Subsequently, all characters that you enter and that MS-DOS displays are printed until you again press Fn-Echo (or Ctrl-P).

NOTE: Pressing Fn-Echo or Ctrl-P when no printer is attached causes your system to appear locked up (i.e., it won't accept any input). To return to normal operation, simply press Fn-Echo or Ctrl-P again.

Halting Execution of a Command

To halt execution of a command you have entered, press Fn-Break (or Ctrl-C). (This is often called *aborting* a command.) For example, suppose you enter the following command to print the contents of 1QSALES:

```
A > TYPE  1QSALES > PRN
```

To halt printing, simply press Fn-Break (or Ctrl-C).

Pressing Fn-Break (or Ctrl-C) also lets you exit from many application programs.

NOTE: If there are any keys in the keyboard buffer (you've typed ahead), you must use Fn-Break to halt MS-DOS command execution; Ctrl-C won't work in this case.

Command Filters

A filter is a command that reads your input, transforms it according to your instructions, and sends it to a device or file you specify. In this way, the program filters the data. Since command filters can be put together in many different ways, you can use a few filters rather than a large number of specific commands.

MS-DOS command filters include FIND, MORE, and SORT. Their functions are described below:

FIND	Searches for a string of characters in a file.
MORE	Takes standard output and displays it one screen at a time.
SORT	Arranges data in alphabetical and numerical order. The order can be either in ascending (0 to 9, A to Z) sequence or in descending (9 to 0, Z to A) sequence.

The next section briefly describes how to use these filters. They are described in more detail in Chapter 7.

Command Piping

Command piping is the process of giving more than one command to the system at a time and making the commands work together. Piping is useful when you want to use the output of one program as the input to another program.

You perform piping by separating commands with the pipe separator, which is the vertical bar symbol (|). (Press Fn-Shift-| to enter the | character.) For example, the command

```
A>DIR | SORT
```

provides an alphabetically sorted listing of a directory. The vertical bar causes all output generated by the left side of the bar to be sent to the right side of the bar for processing.

You can also use piping when you want to send output to a file. If you want your directory sorted and sent to a new file, you could type:

```
A>DIR | SORT>DIREC.FIL
```

MS-DOS creates the file DIREC.FIL on your default drive. DIREC.FIL contains a sorted listing of the directory on the default drive because no other drive was specified in the command. To specify a drive other than the default drive, you could type:

```
A>DIR | SORT>B:DIREC.FIL
```

This sends the sorted data to the file DIREC.FIL on drive B:.

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

```
A>DIR | SORT | MORE
```

The MORE filter displays the sorted directory one screen at a time, and places the following message at the bottom of each screen:

```
—MORE—
```

which indicates more output remains. Press any key to display the next screen.

Chapter 7: MS-DOS Commands

This chapter has two parts. The first part describes MS-DOS file handling and utility commands, and includes the following topics:

- □ Command summary
- □ Conventions used to describe the commands
- □ General rules for entering commands
- □ File specifications—the representation of a specific file to a command: the device, the directory, the name of the file, the filename extension, and wildcard characters. (Chapter 2, The File Specification, treats these topics in detail.)

The second part contains detailed descriptions of each command. They are organized in alphabetical order.

External and Internal Commands

There are two types of MS-DOS commands:

- □ Internal commands
- □ External commands

Internal commands are in the COMMAND.COM file (the command processor) that is loaded into main memory (RAM) along with MS-DOS.

External commands exist as separate files; the external command and the corresponding filename are the same. The file corresponding to the external command must be present on an active device in order for it to execute.

Command Summary

Table 7-1 both summarizes the commands described in this chapter and provides examples of how they are commonly used. The sections following Table 7-1 give details on the rules and formats of the commands.

Table 7-1. *Summary of MS-DOS Commands*

Command	**Result**
APPEND	Sets a search path for data files.*
ASSIGN	Changes the device letter MS-DOS assigns to a particular device.*
ATTRIB	Sets or displays file attributes.*
BACKUP	Backs up files from a hard disk to diskettes.*
BREAK	Sets Ctrl-C check.
CD (or CHDIR)	Displays the name of the current directory.
CD \SALES	Makes SALES the current directory.
CD ..	Makes the next highest directory the current directory.
CD \	Makes the root directory the current directory.
CHKDSK C:/F	Gives a status report on the hard disk and repairs any damaged files found.*
CLS	Clears the screen.
COPY source target	Copies the data in the file named *source* to a file named *target*. Any data in *target* is overwritten; if *target* doesn't already exist, MS-DOS creates it.
COPY memo1.bak A:	Copies *memo1.bak* on the default device to a file named *memo1.bak* on the diskette in drive A.

*External command; the corresponding command file must be on a storage device connected to your computer.

Command	Result
COPY memo?.bak A:	Copies all files on the default device that have *memo* for the first four characters in the filename, any valid character (or no character) for the fifth, and a *.bak* extension, to drive A. (MEMO.BAK is also copied, if it exists.)
COPY *.bak A:	Copies all files on the default device that have the extension *.bak* to the diskette in drive A; the filename can consist of one or more valid characters, as designated by the wildcard (*). The target files will have the same names as the source files.
COPY source A:target	Copies *source* on the default device to a file named *target* on the diskette in drive A.
COPY *.* A:	Copies all files in the current directory of the default device to the diskette in drive A. The filename and extension can consist of one or more valid characters, as designated by the wildcards (*).
COPY C:*.* A:	Copies all files in the current directory of drive C: to the diskette in drive A.
COPY old1 + old2 new	Copies data from *old1* and *old2* and puts it into *new*.
CTTY	Changes the device from which you issue commands.
DATE	Displays the date in the computer calendar and lets you change it.
DATE 4-25-86	Changes the date in the computer calendar to April 25th, 1986.
DEL memo1.add	Deletes the file *memo1.add* on the default device.

*External command; the corresponding command file must be on a storage device connected to your computer.

❯

Command	Result
DEL memo?.add	Deletes matching files in the current directory on the default device. The first four characters of the filename must be *memo*; the fifth character can be any valid character or no character; and the extension must be *.add*.
DEL *.add	Deletes all files in the current directory on the default device that have the extension *.add*. The filename can consist of one or more valid characters, as designated by the wildcard (*).
DEL A:sales	Deletes the file *sales* on the diskette in drive A.
DEL C:sales	Deletes the file *sales* in drive C.
DEL A:*.*	Deletes all files on the diskette in drive A. The filename and extension can consist of one or more valid characters, as designated by the wildcards (*).
DIR	Displays the name, size, and creation or modification dates of the files in the current directory.
DIR /W	Displays the directory in a multi-column format the width of the screen.
DIR /P	Displays a multi-page directory a screen at a time. You skip to each new screen by pressing any key.
DIR /W/P	A combination of DIR/W and DIR/P.
DIR > PRN	Prints directory information.
DIR SALES	Displays directory information on all files named SALES, regardless of the extension.
DIR .TXT	Displays directory information on all files with the extension TXT, regardless of the filename.
DISKCOMP	Compares the contents of two diskettes (sector by sector).*

*External command; the corresponding command file must be on a storage device connected to your computer.

Command	Result
DISKCOPY A: A:	Copies all data from a source diskette in drive A to a target diskette, also in drive A. MS-DOS prompts you when to insert the appropriate diskette. If necessary, DISKCOPY initializes the target diskette to the MS-DOS format and verifies that it has the same number of sectors as the source diskette. **CAUTION:** You cannot copy a 3½-inch diskette to a 5¼-inch diskette with this command. Use the COPY or XCOPY commands instead.*
DISKCOPY A: B:	Copies all data from the diskette in drive A to the diskette in drive B. If necessary, DISKCOPY initializes the target diskette to the MS-DOS format and makes sure that it has the same number of sectors as the source diskette. (See the cautionary note on the previous DISKCOPY example in this table.)*
EXE2BIN	Converts executable files (.EXE) to binary format.*
EXIT	Exits the command processor (COMMAND.COM) and returns to a previous level, if one exists.
FDISK	Partitions a hard disk for use as an MS-DOS storage device; displays information on hard disk storage devices; changes the active status of a partition and deletes a DOS partition. (See Chapter 4 for more information.)*
FIND	Searches for a specific string of text in a file or files.*
FORMAT A:/S	Initializes the diskette in drive A and copies the two MS-DOS system files and the COMMAND.COM file onto it; erases any existing files on the diskette.*
FORMAT B:	Initializes the diskette in drive B.*

*External command; the corresponding command file must be on a storage device connected to your computer.

❯

Command	Result
GRAFTABL	Loads a table of graphics characters into main memory (RAM).*
GRAPHICS	Activates the graphics print-screen facility for graphics screen images when the display is in graphics mode.*
HELP	Provides on-screen information about MS-DOS commands.*
JOIN	Joins a disk drive to a specific path.*
LABEL	Creates, changes, or deletes disk volume labels.*
MKDIR \EAST or MD \EAST	Creates a directory named EAST subordinate to the root directory.
MD WEST	Creates a directory named WEST subordinate to the current (working) directory.
MD \EAST\SALES	Creates a directory under EAST named SALES.
MODE	Sets options for controlling output devices.*
MODE MODEM = OFF	Turns off the power to the optional internal modem.*
MODE MODEM = ON	Turns on the power to the optional internal modem, at the same time turning off the power to the serial port if it was on.*
MODE SERIAL = OFF	Turns off the power to the serial port.*
MODE SERIAL = ON	Turns on the power to the serial port; at the same time it turns off the power to the optional internal modem if it was on.*
MORE	Sends output to the screen one page at a time.*

*External command; the corresponding command file must be on a storage device connected to your computer.

Command	Result
PATH	Displays or sets a command search path.
PCSLAVE	Operates with the PCMASTER device driver to permit communication between a GRiDCase and an IBM PC or PC/XT.*
PRINT	Prints a text file on a printer while you are processing other MS-DOS commands.*
PROMPT	Changes the MS-DOS command prompt.
RECOVER	Recovers a file or an entire disk containing bad sectors.*
REN oldname newname	Changes the name of the file *oldname* to *newname*. The file is on the default device.
REN A:oldname newname	Changes the name of the file *oldname* to *newname*. The file is on the diskette in drive A.
RMDIR \EAST or RD \EAST	Removes a directory named EAST subordinate to the root directory. The subdirectory must be empty before you can remove it.
RD WEST	Removes a directory named WEST subordinate to the current (working) directory. The subdirectory must be empty before you can remove it.
REPLACE	Copies newer versions of files over previous versions.*
RESTORE	Restores one or more files from backup diskettes to a hard disk.*
SCANCASE	Displays information on the components that make up the configuration of the GRiDCase on which the command is issued.*

*External command; the corresponding command file must be on a storage device connected to your computer.

❯

Command	Result
SET	Sets one string value equivalent to another string.
SHARE	Installs file sharing and locking for networks.*
SORT	Reads input from the keyboard or a device, sorts it, then writes it to the screen or a file.*
SUBST	Substitutes a string for a pathname.*
SUBST X: C: \MKT \WEST \NEW	Substitutes the string X: for the pathname C: \MKT \WEST \NEW. You can now get to this directory by typing only X:*
SYS A:	Copies MS-DOS system files from the default device to the diskette in drive A.*
TIME	Displays the time of the computer clock and lets you change it.
TIME 8:30	Changes the time of the computer clock to 8:30 a.m.
TREE C:	Displays the path (and optionally lists the contents) of each directory on drive C.*
TYPE	Displays the contents of a file on the screen.
VER	Displays the version number of the MS-DOS system installed in your computer.
VERIFY	Controls the verify switch when writing to a disk.
VOL	Displays the disk volume label, if it exists.
XCOPY	Copies files and directories, including lower level directories if they exist.*
XCOPY A: B: /S /E	Copies all files and subdirectories on the disk in drive A to the disk in drive B.*

*External command; the corresponding command file must be on a storage device connected to your computer.

Conventions Used in Command Descriptions

This chapter presents command descriptions in the following format:

COPY filespec1 [filespec2] [/V] [/A] [/B]

Note the following rules that apply to this format:

- □ All words shown in capital letters—COPY in the above example—are commands. You must enter them *exactly* as shown, though you can use any combination of uppercase or lowercase characters.
- □ You supply the text for items shown in lowercase letters. In the above example, you enter your own text to specify *filespec1* (the file from which data is to be copied) and *filespec2* (the file to which the data is to be copied).
- □ Items in brackets ([ ]) are optional. If you include optional information, don't enter the brackets, only the information within them.
- □ Some commands have *switches* that allow you to vary the way a command performs its functions. A switch is identified in the command description by a slash followed by an alphanumeric character—*/V*, */A*, and */B* in the above example; you can enter the character in either uppercase or lowercase.
- □ An ellipsis (. . .) indicates that you can repeat an item as many times as you want.
- □ With the exception of square brackets, you must include all punctuation where shown, such as commas, equal signs, question marks, colons, slashes (/), or backslashes (\).
- □ A vertical bar (|) that separates items indicates that you must choose one of the options listed.

General Rules

The following general rules apply to MS-DOS commands.

- □ Commands and the text you supply with them can be entered in either uppercase or lowercase characters.
- □ If you want to use an external command that exists on a different drive or directory, simply precede the command with the drive and pathname where it exists.
- □ Commands are often followed by accompanying text called *parameters*. For example, a parameter can be the name of a file that you want to duplicate with the COPY command.

- □ Delimiters separate commands from their parameters. A delimiter can be a space, comma, semicolon, equal sign, or the tab character. A space is the most common delimiter; it is used in the following examples and throughout this manual.

```
DEL MYFILE.OLD NEWFILE.TXT
RENAME THISFILE THATFILE
```

- □ Don't insert spaces in file specifications that already contain special characters that delimit their parts. In the following example, the colon serves as a delimiter; consequently, a space between the device letter and the filename would cause an error.

```
COPY A:OLDFILE C:NEWFILE
```

- □ When instructions say "Press any key," you can press Tab, Esc, Return, an arrow key, the space bar, or any alpha (A to Z) or numeric (0 to 9) key.
- □ Include the filename extension when referring to a file that has an extension.
- □ Press Ctrl-C or Fn-Break to halt execution of a command.
- □ Commands start execution only after you have pressed Return.
- □ Wildcards (the characters * and *?*) and device names (for example, PRN or CON) are not allowed in the names of any commands.
- □ When commands produce a large amount of output on the screen, the text automatically scrolls forward. You can press Fn-Pause or Ctrl-S to stop scrolling. Press any key to resume scrolling.
- □ Diskette drives, hard disk drives, and other storage devices are referred to as source drives and target drives. A source drive is the drive from which you transfer information. A target drive is the drive to which you transfer information.
- □ There are several special MS-DOS editing keys available which allow you to recall and edit the last MS-DOS command you entered. The last MS-DOS command entered is always stored in a special area called the template, where it is available for recall. These editing keys are described in Chapter 9 in Table 9-1.

NOTE: The following commands do not work over a network (on a shared or remote device): CHKDSK, DISKCOMP, DISKCOPY, FORMAT, LABEL, RECOVER, SUBST, and SYS.

APPEND

The APPEND command sets a search path for data files.

Type: External

APPEND has the following format:

APPEND [d:][path][;[d:][path] . . .]

where *d:* and *path* specify the drive and directory that MS-DOS searches for data files. You can specify more than one path to search by separating each path with a semicolon (;).

If you use the APPEND command with no options, the current data path is displayed. The command

```
APPEND;
```

sets the nul data path. This means that MS-DOS searches only the working directory for data files.

Append searches the data path for all files, regardless of their files extensions, only with the following MS-DOS system calls:

0FH	Open file (FCB)
23H	Get file size
3DH	Open handle

You can use the APPEND command across a network to locate remote data files.

Example

The following command sets a search path for data files to the \WORDPROC \MEMOS and the \MEMOS directories on drive C:

```
APPEND C:\WORDPROC\MEMOS;C:\MEMOS
```

This means that first MS-DOS will search the working directory for any data files, then the C: \WORDPROC \MEMOS directory, then the C: \MEMOS directory.

ASSIGN The ASSIGN command changes the device letter that MS-DOS automatically assigns to a particular device. This command is useful if your application program requires that you put data files on diskettes, and you prefer to use some other storage medium—for example, a hard disk.

Type: External

ASSIGN has the following format:

ASSIGN [d1 [=] d2] [. . .]

where *d1* is the letter of the device to be re-assigned and *d2* is the letter of the device to be used in place of *d1*. If you omit both *d1* and *d2* , MS-DOS cancels any previously made assignments. The equal sign is optional.

CAUTION: Immediately cancel an assignment when you no longer need it, by simply entering ASSIGN. If you do not cancel assignments, you can unintentionally delete files and lose data.

You can make multiple assignments in the same command by separating each assignment with a comma or space as shown in the following example:

```
ASSIGN A=C B=D
```

A new assignment remains in effect for all commands that refer to that device until you cancel it.

NOTE: You should not use the ASSIGN command with the BACKUP or PRINT commands, since these commands require actual drive information.

Example

The following is an example of the ASSIGN command:

```
ASSIGN A=C
```

The above command tells MS-DOS to assign all requests that access drive A to drive C. Subsequently, all commands using the device letter A are referred to drive C. For example, the following command lists the working directory of drive C:

```
DIR A:
```

ATTRIB

The ATTRIB command sets or displays the read-only and archive attributes of a file. This command is useful for protecting files that are to be shared by making them read-only files. In addition, when used with the BACKUP, RESTORE, and XCOPY commands, you can selectively copy only those files marked with the archive attribute.

Type: External

The ATTRIB command has the following format:

```
ATTRIB [+r | -r] [+a | -a] [filespec]
```

where: *+r* sets the read-only attribute of a file
–r returns the file to read or write status
+a sets the archive attribute of a file
–a clears the archive attribute of a file

and *filespec* specifies the file to set or display attributes for.

You can use the ATTRIB command to make a file read-only for sharing over a network.

The BACKUP, RESTORE, and XCOPY commands use the archive attribute to control selective copying of files. The archive attribute allows you to make copies of only those files that have changed. The archive attribute is automatically set when a file is modified. For unmodified files, you can manually set the archive attribute using the ATTRIB command. Then you can selectively backup files using the /M switch (see BACKUP and RESTORE), or copy files using the /A or /M switches (see XCOPY).

To display a file's attributes, specify only the filename as follows:

```
ATTRIB C:REPORTS\JUNE
```

The wildcard characters (*.*) can be used to print the attributes of all files on a specific device.

Examples

The following example makes the file named FORECAST.TXT read-only:

```
ATTRIB +R C:FORECAST.TXT
```

The following example sets the archive attribute for all files with the extension .TXT in the specified directory:

```
ATTRIB +A C:NEWBOOK\*.TXT
```

BACKUP

The BACKUP command backs up files from one disk to another. When used to backup a hard disk, the BACKUP command lets you sequentially store files on one or more diskettes. The files are not immediately accessible; you must use the RESTORE command to return the files to the original disk to make them usable.

Type: External

The BACKUP command has the following format:

```
BACKUP d:[pathname] d: [/S] [/M] [/A] [/D:mm-dd-yy] [/T:hh:mm]
[/L:filename]
```

where the first *d:* is the disk drive that you want to back up and the second *d:* is the backup disk drive. If you don't specify *pathname*, the files from the current directory are backed up. If you don't specify a filename, all files in the directory are backed up.

BACKUP can back up files on disks of different media. The media do not have to have the same number of sides or sectors. For example, you can back up files from:

Hard disk to floppy disk.
Floppy disk to floppy disk.
Floppy disk to hard disk.
Hard disk to hard disk.

If the target drive is a floppy disk, BACKUP places the backed up files in the root directory. If the target drive is a hard disk, the backed up files are placed in a subdirectory named BACKUP. You must format new disks before BACKUP can use them.

NOTE: Before you start the BACKUP command to back up a hard disk onto floppy disks, be sure you have enough formatted diskettes available to hold all the files you want to back up. It could require nearly 15 three and a half inch diskettes to back up a full 10MB hard disk.

You can use a wildcard character in the filename to back up all files that match the specified filename. For example,

```
C>BACKUP C:*.COM A:
```

backs up all files from the current directory of the hard disk drive (Drive C) with the extension .COM onto the diskette in Drive A.

The following switches are provided with BACKUP:

/S — Backs up all files at all levels of subdirectories, as well as the files in the specified directory.

/M — Backs up only files that have been modified since the last backup procedure. This can save time because it avoids backing up files that never change.

/A — Adds backup files to a backup diskette in the specified drive. If you don't use the /A option, MS-DOS erases the existing backup files on the diskette, except for hidden files.

/D:mm-dd-yy — Backs up files that were modified on or after the specified date.

/T:hh:mm — Backs up only those files that were modified at or after the specified time.

/L:filename — Makes a backup log entry in the specified file. If you do not give a filename, the entries are placed in a file named BACKUP.LOG in the root directory of the files being backed up. The first line in each entry contains the date and time of the backup. Each subsequent line in the entry contains the filename of a backed up file and the number of the floppy disk that contains the file. You can use this information to restore a particular file from a floppy disk. If the backup log already exists, the entry is appended to the end of the log. (**CAUTION:** IBM BACKUP/RESTORE compatibility may be lost if you use this switch.)

The BACKUP command displays the name of each file as it backs it up. You can print this list by redirecting the output to the printer. See Chapter 6 for detailed information on redirecting output. You should label and number each backup disk consecutively to help you restore the files properly with the RESTORE command. If you're backing up hard disk files onto floppy disks and you have multiple hard disks, be sure to indicate on the backup diskettes the hard disk they were made from.

CAUTION: You should not use the BACKUP command if the drive you are backing up has been assigned, joined, or substituted. If you do, you may not be able to restore the files with the RESTORE command.

Examples

The following example backs up all files that are on the hard disk (Drive C):

```
C>BACKUP C:\ A: /S
```

The following example backs up two different files from the hard disk onto the same backup diskette:

```
C>BACKUP C:\EMPLOYEE\CURRENT A:
C>BACKUP C:\EMPLOYEE\FORMER A: /A
```

The following example backs up all files in the current directory that have changed since the last backup procedure:

```
C>BACKUP C:*.* A: /M
```

Using the BACKUP Command with Batch Files

When you complete a backup program within a batch file, you can test for the following exit codes with the IF command.

Code	Status
0	Normal completion
1	No files found to backup
2	Some files not backed up due to sharing conflicts
3	Terminated by user
4	Terminated due to error

BREAK

The BREAK command controls subsequent MS-DOS operations after you press Ctrl-C.

Type: Internal

BREAK has the following format:

```
BREAK [ON | OFF]
```

Depending on the program you are running, Ctrl-C may be used to stop an activity (for example, to stop sorting a file). Normally, MS-DOS checks to see if Ctrl-C has been typed while it is reading from the keyboard, writing to the screen, or to a printer. Setting BREAK to ON allows Ctrl-C checking to be extended to other functions such as disk reads and writes.

To check for Ctrl-C during screen, keyboard, and printer operations only, set BREAK to OFF.

To find out how BREAK is currently set, simply type BREAK.

CHDIR

The CHDIR (or CD, for *change directory*) command either changes your current directory or displays the name of your current directory.

Type: Internal

CHDIR has either of the following formats:

```
CHDIR [pathname]
CD [pathname]
```

where *pathname* is one or more subdirectory names that specify the new current directory.

For example, the command

```
CD \SALES\EAST\EXPENSES
```

changes the current directory to EXPENSES. The short-hand notation .. (two periods) used with the command changes the current directory to the parent directory (the directory immediately above the current directory). For example, after the above command,

```
CD ..
```

changes the current directory to EAST. To find out what directory you're currently using, enter

```
CD
```

Using the above example, MS-DOS displays this pathname:

```
\SALES\EAST
```

To return directly to the root directory, enter

```
CD \
```

CHKDSK

The CHKDSK (for *check disk*) command checks for errors in a disk and, if specified, attempts to repair any errors it finds.

Type: External

CHKDSK has the following format:

CHKDSK [d:[filespec]] [/F] [/V]

where *d:* is the drive letter of the disk to be checked. If you don't specify a device, the default device is checked.

CHKDSK displays a status report showing current usage of main memory (RAM) and the storage media specified in the command. Here's an example of the status report you see when the command executes:

```
730112 bytes total disk space
 28672 bytes in 2 hidden files
229376 bytes in 29 user files
472064 bytes available on disk
524288 bytes total memory
494752 bytes free
```

The item *disk space* in the status report can refer to a diskette, hard disk, or other storage device, whichever you specify (see below); *hidden files* refer to the MS-DOS system files that constitute the operating system; *directories* and *user files* show the space occupied by directories and files that you create; *memory* refers to the amount of RAM in your computer.

If you include *filespec*, CHKDSK reports on whether the specified file(s) occupies contiguous space on the disk, or if it is split into two or more parts in different disk locations.

The following switches are provided with CHKDSK:

/F CHKDSK repairs (*F* stands for *fix*), as well as reports the errors it finds.

/V CHKDSK displays informational messages (in addition to the status report) as it executes (*V* stands for *verbose*).

You can cause the output of CHKDSK to be printed or stored in a file by redirecting the output with >PRN or >filespec, respectively.

NOTE: Don't specify /F when redirecting the CHKDSK output.

CLS

The CLS (for *clear screen*) command erases all the information (except for the system prompt) from your display screen.

Type: Internal

CLS has the following format:

```
CLS
```

COMMAND

The COMMAND command starts the command processor.

Type: External

COMMAND has the following format:

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

This command starts a new command processor (the MS-DOS program that contains all internal commands).

The command processor is loaded into memory in two parts: the *transient* part and the *resident* part. Some application programs write over the transient part of COMMAND.COM when they run. When this happens, the resident part of the command processor looks for the COMMAND.COM file on the disk so it can reload the transient part.

The *d:pathname* option tells the command processor where to look for the COMMAND.COM file if it needs to reload the transient part into memory. The *cttydev* option allows you to specify a different device for input and output. See the CTTY command for more information.

The following switches are provided with COMMAND:

/P	Tells COMMAND.COM not to exit to any higher level.
/E:nnnnn	Specifies the environment size, where *nnnnn* is the size in bytes. Valid sizes range between 160 and 32768 bytes. The default value is 160 bytes. If *nnnnn* is less than 160 bytes, MS-DOS defaults to 160 bytes and gives the message: "Invalid environment size specified." If *nnnnn* is greater than 32768 bytes, MS-DOS gives the same message and defaults to 32768 bytes.
/C string	Tells the command processor to execute the command or commands specified by *string* and then return to the next higher level.

The following example tells the command processor to start a new command processor under the current program; run the command CHKDSK B:; and return to the first command processor.

```
COMMAND /C CHKDSK B:
```

COPY

The COPY command duplicates a *source file* or a group of source files to one or more *target files*. This command also appends files.

Type: Internal

The COPY command has the following format when you copy (rather than add) data to a file:

```
COPY filespec1 [filespec2] [/V] [/A] [/B]
```

where *filespec1* is the source file and *filespec2* is the target file; you can specify either an existing file or a new file for *filespec2* .

The following switches are provided with COPY:

/V Verifies that data is recorded correctly on the target device. This increases the time it takes to copy a file, so use it only in a second copy operation when the first copy attempt fails—for example, if you can't get to the data in the target file after the first copy operation. Specifying /V can help isolate the cause of the problem. This switch is also useful in making a backup of a hard disk before you reformat it.

/A When used with a source filespec, causes the file to be treated as an ASCII (text) file. Data in the file is copied up to but not including the first end-of-file mark. The remainder of the file is not copied.

When used with a destination filespec, causes an end-of-file character to be added as the last character of the file.

/B When used with a source filespec, causes the file to be treated as a binary file. The entire file is copied, including any end-of-file characters.

When used with a destination filespec, causes no end-of-file character to be added.

Note that the /A and /B switches apply to the file specification that precedes them and to all remaining file specifications in the command, until another /A or /B is encountered.

When appending files, the default switch is always /A.

When you use the COPY command, note the following:

□ If the source and target files are in the current directory, you don't need to specify a device letter or directory path. For example, the command

```
C>COPY REPORT.TXT REPORT.BAK
```

copies the data in REPORT.TXT to REPORT.BAK; both files are within the same directory.

□ When you specify wildcards in order to copy an entire directory of files, COPY ignores subdirectories and their contents. For example, the following command

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

copies only the files from the current directory on drive A to the current directory on drive C; files in subdirectories on drive A are *not* copied. You must use the XCOPY command if you want to copy all subdirectories also.

□ If the source file (*filespec1*) is on the default drive and the target file (*filespec2*) is not specified, MS-DOS displays an error message; you can't copy a file to itself. For example, the following command is in error:

```
C>COPY REPORT.TXT
```

□ If you don't specify a target file (*filespec2*), MS-DOS assumes the target file is on the default drive and gives it the same name as the source file (*filespec1*). For example, the command

```
C>COPY A:REPORT.TXT
```

copies the data in REPORT.TXT located on diskette to the file REPORT.TXT located in the default directory on the default drive (drive C:).

□ If the target file (*filespec2*) is a device letter (d:) only, the target file is given the same name as the source file.For example, the command

```
C>COPY A:REPORT.TXT B:
```

copies the data in REPORT.TXT on disk drive *A* to the file REPORT.TXT on disk drive *B*.

Concatenating Files

Concatenating files is the process of joining two or more source files into one target file. The COPY command permits concatenation; simply separate the source files in a COPY statement with plus signs (+). For example, the command

```
COPY A.XYZ + B.BAK + C.TXT BIGFILE.TXT
```

joins files A.XYZ, B.BAK, and C.TXT and places them in BIGFILE.TXT.

To combine several files using wildcards into one file, you can type:

```
COPY *.LST COMBIN.PRN
```

This command joins all files with the extension .LST into a file named COMBIN.PRN.

In the next example, each file that matches *.LST, is combined with the corresponding .REF file. The result is a file with the same filename but with the extension .PRN. Thus, FILE1.LST is combined with FILE1.REF to form FILE1.PRN; then XYZ.LST with XYZ.REF to form XYZ.PRN; and so on.

```
COPY *.LST + *.REF *.PRN
```

The next example combines all files matching *.LST, then all files matching *.REF, into one file named COMBIN.PRN:

```
COPY *.LST + *.REF COMBIN.PRN
```

COPY compares the filename of the input file with the filename of the target file. If they are the same, that one input file is skipped, and the error message "Content of destination lost before copy" is displayed. Further concatenation proceeds normally. This allows "summing" files, as in this example:

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

This command appends all *.LST files, except ALL.LST itself, to ALL.LST. This command executes successfully, even though it issues an error message.

Using COPY with Pathnames

Using pathnames with the COPY command is an easy way to duplicate all files in a subdirectory. For example:

```
C>COPY SALES\*.* \TAXES
```

copies all files in the subdirectory SALES to the subdirectory TAXES.

```
C>COPY \SALES \1986
```

copies all files in the subdirectory SALES to the subdirectory 1986.

```
C>COPY SALES\*.TXT A:\WP
```

copies *.TXT files in the subdirectory SALES to the subdirectory WP on drive A.

CTTY

The CTTY command changes the device from which you issue commands.

Type: Internal

CTTY has the following format:

CTTY d:

where *d:* is the device from which you are giving commands to MS-DOS. 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 keyboard) to the AUX port. The command

```
CTTY CON
```

returns command I/O to the console.

DATE

The DATE command displays the current date as recorded by the internal clock of your computer and lets you change it, if you wish. MS-DOS handles leap years and sets the correct date in the transition from one month to the next, regardless of the number of days (28, 29, 30, or 31) in the preceding month.

The date you set determines the date you see in a file directory that notes when a file was created or last modified.

Type: Internal

DATE has the following format:

```
DATE [mm-dd-yy[yy]]
```

Entering the command DATE displays this information:

```
Current date is mm-dd-yy
Enter new date (mm-dd-yy):_
```

Press Return if you don't want to change the date shown, or type in a new date in the format mm-dd-yy, where *mm* is a number from 1 through 12 specifying the month, *dd* is a number 1 through 31 specifying the day, *yy* is a number from 80 through 99 specifying the year. Optionally, a four character year may be entered from 1980 to 2099. For example, the date July 1st, 1986 would be entered as follows:

```
7-1-86
```

You can bypass the prompt by entering the command followed by the date as shown below:

```
DATE 7-1-86
```

Separate the date, month, and year by hyphens (-) or slashes (/).

DEL

The DEL (for *delete*) command deletes a file, or, when using wildcards, a group of files from a directory.

Type: Internal

DEL has either of the following formats:

DEL *filespec*
ERASE *filespec*

where *filespec* specifies the file(s) to be deleted.

Specifying the wildcards *.* refers to all files in the current directory. For example, the following command deletes all the files in the current directory:

```
DEL *.*
```

However, before this command executes, the prompt

```
Are you sure (Y,N)?
```

appears. If you type Y (for yes), then all files are deleted as requested.

DEVICES

The DEVICES command displays a report of all the devices recognized by the system, including devices installed in the CONFIG.SYS file such as RAMDRIVE. Slave devices are also included if the PCMASTER.SYS statement is included in the CONFIG.SYS file. Devices installed in the CONFIG.SYS file are referred to as "user-installed" devices.

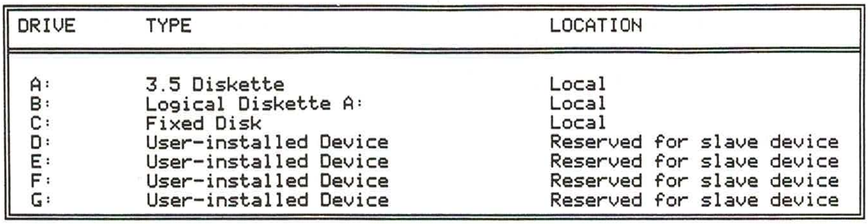
Type: External

DEVICES has the following format:

```
DEVICES
```

Note: If the display is in 40 column mode when the DEVICES command is used, it is automatically switched to 80 column mode for the DEVICES display. It is not converted back to 40 column mode when DEVICES is finished.

A typical DEVICES display is shown below.



```
DRIVE      TYPE                                LOCATION

  A:       3.5 Diskette                        Local
  B:       Logical Diskette A:                 Local
  C:       Fixed Disk                          Local
  D:       User-installed Device               Reserved for slave device
  E:       User-installed Device               Reserved for slave device
  F:       User-installed Device               Reserved for slave device
  G:       User-installed Device               Reserved for slave device
```

DIR

The DIR (for *directory*) command lists the files residing on the device and directory you specify.

Type: Internal

DIR has the following format:

DIR [filespec] [/P] [/W]

Note the following:

- □ Typing only DIR without the optional parameters displays a directory of all the files in the current directory.
- □ Typing DIR followed by a device letter (DIR d:), displays a directory of all the files on the specified device.
- □ Typing DIR followed by a filename without an extension (DIR filename) displays a directory of all files with the same name, regardless of extension, in the current directory.
- □ Typing DIR followed by a device letter and a filename with an extension (for example, DIR d:filename.ext), displays directory information on the file specified.

You can include the wildcards ? and * (question mark and asterisk) as part of the filename. For your convenience, the following DIR commands are equivalent:

Command	Equivalent
DIR	DIR *.*
DIR FILENAME	DIR FILENAME.*
DIR .EXT	DIR *.EXT

The following switches are provided with DIR:

/P Displays the directory one screen at a time. You can go from screen to screen, through the directory, by pressing any key.

/W Displays only filenames (in several columns) without the size, date, and time information.

DISKCOMP

The DISKCOMP (for *disk compare*) command compares the contents of two diskettes. In contrast, the File Comparison (FC) utility (described in Chapter 10), compares two specific files on a diskette or other storage medium.

DISKCOMP issues a message when a mismatch occurs, indicating the track number (0 to 79 for a 3½-inch diskette or 0 to 39 for a 5¼-inch diskette), and the side of the diskette (0 or 1) where it occurred.

You would normally use DISKCOMP to determine if a target diskette was copied correctly after executing the DISKCOPY command, or to verify if two diskettes are the same.

Type: External

DISKCOMP has the following format:

```
DISKCOMP [d1:] [d2:] [/1] [/8]
```

where *d1* and *d2* are device letters that indicate the drives on which the two diskettes to be compared reside.

The following rules apply to the device letters.

- □ If you don't specify device letters, DISKCOMP assumes that the source diskette and the diskette to be compared are in the default drive; it then prompts you to insert each diskette when required. For example, the command

  ```
  A>DISKCOMP
  ```

  prompts you to use drive A for both the source diskette and the diskette to be compared.
- □ If you have only one drive, MS-DOS prompts you to use drive A (logical drive B:) regardless of what you specify for *d1* and *d2*.

The following switches are provided with DISKCOMP:

/1 DISKCOMP compares only the first side of a two-sided diskette. Otherwise, DISKCOMP determines the number of sides to compare based on the diskette in drive *d1*.

/8 DISKCOMP compares only 8 sectors per track, regardless of the number of sectors the first diskette contains. Many older diskettes are formatted 8 sectors per track. Use this option when you compare data that was copied from such a diskette.

Note the following:

- □ You cannot use DISKCOMP to compare a 3½-inch diskette with a 5¼-inch diskette. The difference in the the number of tracks on each diskette produces an error during the comparison process. Instead, use the FC utility described in Chapter 10; FC compares only the files you specify rather than the entire contents of the two diskettes.
- □ If the source diskette is two-sided, DISKCOMP tries to compare both sides. If the target diskette is one-sided, a compare error occurs, unless you specify the /1 option, described above.
- □ If the source diskette drive is one-sided, DISKCOMP compares only one side.
- □ If you are comparing a disk with a copy of the disk made by the COPY command, DISKCOMP may give a "Compare error" message, even if the files on the disks are identical. This is because the COPY command duplicates the information, but it doesn't necessarily place the information in the same location on the target disk. In this case, you should use the FC utility, described in Chapter 10, to compare individual files on the disks.
- □ You cannot use DISKCOMP with assigned, joined, or substituted drives, or over a network.

Examples

The following example compares the contents of two diskettes in drive A. You are prompted to insert the second diskette at the appropriate time.

```
A>DISKCOMP
```

The following example compares the first side of the diskette in drive B to the first side of the diskette in the default drive (B: in this example). You are prompted to insert the second diskette at the appropriate time.

```
B>DISKCOMP B: /1
```

Using the DISKCOMP Command with Batch Files

When you complete a DISKCOMP command within a batch file, you can test for the following exit codes with the IF command.

Code	Status
0	Disks compared exactly
1	Disks did not compare
2	Terminated by user (Ctrl-C pressed)
3	Hard error—an unrecoverable read error occurred
4	Initialization error—not enough memory, invalid drives or command syntax

DISKCOPY The DISKCOPY command copies the contents of a source diskette to a target diskette. If you wish to make sure the target diskette was copied correctly, run DISKCOMP after a DISKCOPY operation.

Type: External

DISKCOPY has the following format:

```
DISKCOPY [d1:] [d2:]
```

where *d1* is a letter specifying the device containing the diskette to be copied and *d2* specifies the device containing the target diskette.

After copying, DISKCOPY prompts:

```
Copy another diskette (Y/N)?_
```

If you press Y, another copy operation is performed on the same drives that you originally specified, after you have been prompted to insert the proper disks.

Press N to end the DISKCOPY.

The following rules apply to DISKCOPY:

- □ If you don't specify device letters, DISKCOPY assumes that the source and target diskettes are in the default drive; it then prompts you to insert each diskette when required. For example, the command

  ```
  A>DISKCOPY
  ```

  prompts you to use drive A for both the source diskette and the target diskette.
- □ If you don't specify the second drive letter, the default drive is used as the target drive.
- □ If you have only one drive, MS-DOS prompts you to use drive A (logical drive B:) regardless of what you specify for *d1* and *d2* .
- □ DISKCOPY automatically formats the target diskette to the same format as the source diskette, if necessary.

When To Use COPY or XCOPY Instead of DISKCOPY

Use the COPY or XCOPY commands instead of DISKCOPY in the following instances:

- □ When duplicating files from a 5 ¼-inch diskette to a 3 ½-inch diskette and vice versa because:
  - DISKCOPY causes an error when duplicating files from a 3 ½- to a 5 ¼-inch diskette.
  - DISKCOPY uses space inefficiently on the 3 ½-inch diskette when duplicating from a 5 ¼-inch diskette.
- □ When duplicating an individual file rather than an entire diskette.
- □ When you suspect the source diskette is fragmented. A file is said to be fragmented when, instead of being a separate entity at one location on the diskette, it is split into two or more parts at different locations. Fragmentation occurs when you create, change, or delete files often on the same diskette.

  MS-DOS takes longer to locate, read, and write data in a fragmented file, increasing the time it takes to do your work. The COPY and XCOPY commands compress the fragmented files into a series of contiguous files.

To copy an entire diskette to another diskette, use XCOPY with wildcards as shown below:

```
XCOPY A:*.* B: /S
```

The above example copies all files (except MS-DOS system files) and directories from the diskette in drive A: to the diskette in drive B:.

Using the DISKCOPY Command with Batch Files

When you complete a DISKCOPY command within a batch file, you can test for the following exit codes with the IF command.

Code	Status
0	Copied successfully
1	Non-fatal read or write error
2	Terminated by user (Ctrl-C pressed)
3	Fatal hard error
4	Initialization error—not enough memory, invalid drive or command syntax

EXE2BIN The EXE2BIN command converts .EXE (executable) files to binary format. This saves disk space and loads programs more quickly.

Type: External

EXE2BIN has the following format:

```
EXE2BIN filespec1 [filespec2]
```

This command is useful if you want to convert .EXE files to binary format. *filespec1* designates the input file. If no extension is specified, it defaults to .EXE. The input file is converted to .COM file format (memory image of the program) and placed in the output file specified by *filespec2* . If you don't specify a drive, the drive of the input file is used. If you don't specify an output filename, the input filename is used. If you don't specify a filename extension in the output filename, the new file is given an extension of .BIN.

The input file must be in valid .EXE format produced by the linker. The resident, or actual code and data part of the file must be less than 64K. There must be no STACK segment.

Two kinds of conversions are possible, depending on whether the initial CS:IP (Code Segment:Instruction Pointer) is specified in the .EXE file:

- □ If CS:IP is not specified in the .EXE file, a pure binary conversion is assumed. If segment fixups are necessary (i.e., the program contains instructions requiring segment relocation), you are prompted for the fixup value. This value is the absolute segment at which the program is to be loaded. The resulting program is usable only when loaded at the absolute memory address specified by a user application. The command processor isn't capable of properly loading the program.
- □ If CS:IP is specified as 0000:100H, it is assumed that the file is to be run as a .COM file with the location pointer set at 100H by the assembler statement ORG; the first 100H bytes of the file are deleted. No segment fixups are allowed, as .COM files must be segment relocatable; that is, they must assume the entry conditions explained in the *Macro Assembler Manual*. Once the conversion is complete, you may rename the resulting file with a .COM extension. Then the command processor can load and execute the program in the same way as the .COM programs supplied on your MS-DOS disk.

  In most cases, it is better to use .EXE files rather than .COM files, since .EXE files are relocatable and .COM files are direct memory images that load at a specific memory location and have no room for additional control information.

EXIT

The EXIT command exits COMMAND.COM (the command processor program) and returns to the previous level, if one exists.

Type: Internal

EXIT has the following format:

```
EXIT
```

You can use this command when you are running an application program and want to start the MS-DOS command processor, and then return to your program. After you start the command processor (system call 4BH) you can enter a command:

```
DIR B:
```

and MS-DOS displays the directory for drive B. Typing EXIT next causes you to return to the previous level (your application program).

FIND

The FIND command searches for a specific string of text in a file or files.

Type: External

FIND has the following format:

FIND [/V] [/C] [/N] string [filespec . . .]

FIND is a filter command that takes as options a *string* and a series of *filespecs*. It displays all lines that contain the specified string from the files specified in the command line.

If you don't specify files, FIND takes the input on the screen and displays all lines that contain the specified string.

The following switches are provided with FIND:

/V FIND displays all lines that do not contain the specified string.

/C FIND prints only the count of lines that contain a match in each of the files.

/N FIND precedes each line by its relative line number in the file.

The *string* should be enclosed in double-quotes. For example:

```
FIND ``Fool's Paradise'' BOOK1.TXT BOOK2.TXT
```

displays all lines from BOOK1.TXT and BOOK2.TXT (in that order) that contain the string *Fool's Paradise*.

Insert two double-quotes in the search argument for each double-quote in the file. For example, when searching for *Fool's "Paradise*", enter the command as follows:

```
FIND ``Fool's ````Paradise'''''' BOOK1.TXT BOOK2.TXT
```

The command

```
DIR B: | FIND /V ``DAT''
```

displays all names of the files on the diskette in drive B that don't contain the string DAT.

FORMAT

The FORMAT command initializes storage media—for example, a diskette or hard disk—to a format usable by MS-DOS. You must use this command to format any storage medium that is to hold MS-DOS system files or your data files.

Type: External

FORMAT has the following format:

```
FORMAT d: [/1] [/8] [/B] [/V] [/S]
```

where *d* is a letter specifying a storage device attached to your computer that you want to format. If you specify a hard disk that has been previously formatted and is undamaged, you are prompted with the following message:

```
Enter current Volume Label for drive (x:)
```

Enter the correct volume label to continue formatting your hard disk, or press Return if your hard disk has no volume label. If you give an incorrect hard disk volume label, FORMAT will give an error message and stop.

CAUTION: If essential files already exist on the medium to be formatted, back them up on a diskette or other storage medium. The FORMAT command erases all files on the device being formatted.

The following switches are provided with FORMAT:

/1 Formats only one side of a diskette. Note that if the drive is double-sided and you do not specify this switch, you will not be able to use the disk in a single-sided drive.

/8 Formats a disk with 8 sectors per track. Some older disks are formatted with 8 sectors per track. Use this option when formatting a disk to which an 8-sector disk is to be copied. Normally, FORMAT defaults to 9 or 15 sectors per track, depending on the type of drive used.

/B Formats a disk with 8 sectors per track and allocates space for the hidden system files. If you use this switch, you can place any version of MS-DOS on the disk using that version's SYS command. If you do not use this switch, you can only place MS-DOS version 3.2 on the disk with the SYS command. You cannot use the /V or /S switches if you specify the /B switch.

/V Makes FORMAT prompt you for a volume label to identify the storage medium. A volume label is useful in keeping track of diskettes. It appears in directory listings and status reports produced by DIR and CHKDSK respectively.

You can specify up to 11 characters in the volume label.

/S Makes FORMAT copy the two MS-DOS system files and COMMAND.COM from the default drive to the newly formatted storage medium.

NOTE: The two system files, though present, are hidden and never appear in file directories. To make sure that these files are present, use the CHKDSK (for *check disk*) command described in this chapter.

When formatting is complete, a message is displayed showing the total disk space, any space marked as defective, the total space used by the operating system (when you use the /S switch), and the space available for your files. In addition, the following prompt is displayed:

```
Format another (Y/N)?
```

If you press Y, another format operation is performed on the same drive that you originally specified, after you have been prompted to insert a diskette.

The following table shows which switches you can use for certain types of disks:

Disk Type	Valid Switches
160/180 KB	/1 /8 /B /V /S
320/360 KB	/1 /8 /B /V /S
720 KB	/V /S
1.2 MB	/V /S
hard disk	/V /S

NOTE: You cannot use FORMAT with drives used in the ASSIGN, JOIN, or SUBST commands, or over a network.

Using the FORMAT Command with Batch Files

When you complete a FORMAT command within a batch file, you can test for the following exit codes with the IF command.

Code	Status
0	Successful completion
3	Terminated by user (Ctrl-C pressed)
4	Fatal error
5	"N" response to hard disk prompt, "Proceed with format (Y/N)?"

GRAFTABL

The GRAFTABL command loads additional character data into a table in memory for use with a color or graphics adapter.

Type: External

GRAFTABL has the following format:

```
GRAFTABL
```

After you enter the command, a table of the ASCII characters 128 through 255 is loaded into memory. The message "GRAPHICS CHARACTERS LOADED" is displayed. You can load the graphics table only once each time you start MS-DOS.

If you have a color or graphics adapter, this table lets you display foreign language and special characters when in graphics mode.

GRAPHICS The GRAPHICS command activates the print-screen facility so you can print a graphics display screen when you are using a color or graphics monitor adapter.

Type: External

GRAPHICS has the following format:

```
GRAPHICS [printer] [/R] [/B]
```

where *printer* is one of the following options:

COLOR1	Prints on an IBM Personal Computer Color Printer with a black ribbon.
COLOR4	Prints on an IBM Personal Computer Color Printer with an RGB (red, green, blue, and black) ribbon.
COLOR8	Prints on an IBM Personal Computer Color Printer with a CMY (cyan, magenta, yellow, and black) ribbon.
COMPACT	Prints on an IBM Personal Computer Compact Printer.
GRAPHICS	Prints on an IBM Personal Computer Graphics Printer.

If you do not specify the printer option, *printer* defaults to the GRAPHICS printer type.

The following switches are provided with GRAPHICS:

/R	Prints black and white exactly as seen on the monitor. If you don't specify this switch, the default is to print black as white and white as black.
/B	Prints the background in color. This option is only valid for the COLOR4 and COLOR8 printers.

After you enter the GRAPHICS command, press Fn-PrtSc (the Print Screen key) to print the contents of the screen. Note that the print-screen facility works in text mode without invoking the GRAPHICS command. However, when you are in graphics mode, you must invoke GRAPHICS to ensure a readable printout. GRAPHICS remains in effect until you re-start (boot) your system.

When the screen is in text mode, printing takes less than 30 seconds. When in graphics mode, printing one screen can take as long as three minutes.

NOTE: You can only use the GRAPHICS command with printers that support graphic output, such as dot matrix and inkjet printers.

When in graphics mode, the appearance of the printed copy depends on the graphics resolution, as described below:

Graphics Resolution	**Printed Copy**
320x200 graphics mode	Printed with up to four shades of gray if the printer type is COLOR1 or GRAPHICS.
640x200 graphics mode	Printed sideways. The upper right corner appears in the upper left corner of the printed copy.

HELP

The HELP command provides on-screen information about MS-DOS commands. You can customize HELP to include your own commands or revise the existing topics to suit your specific needs.

Type: External

HELP has the following format:

HELP

When you invoke the HELP command, a list of the topics it covers appears on your screen in alphabetical order, and you see the following prompt at the bottom of the screen:

```
HELP (Enter topic or ESC to exit)
```

You type in the topic or subtopics in the format below and press Return:

TOPIC [subtopic1] [subtopic2] [subtopic3] . . . [subtopic*n*]

The help information concerning the topic or subtopic scrolls onto the screen. A list of subtopics appears, if there are any for the topic you selected. To see the help information concerning a subtopic, type in the subtopic next to the prompt and press Return.

While you are in the HELP command, you can enter a "?" at the prompt to return to the initial list of main topics, or you can simply enter another topic and/or subtopic. Press Esc to return to the MS-DOS prompt.

Customizing Your HELP Files

You can add topics or change the text of your help file to fit your specific needs. Follow the steps below to customize your HELP file.

To Customize the HELP File

1. At the system prompt, enter

   ```
   A>HELP
   ```

   and press Return.

2. Create a HELP.DAT file that you can edit by typing in *E (for *extract*) at the prompt, as follows:

   ```
   HELP (Enter topic or ESC to exit)> *E
   ```

and press Return. After a minute or two, a message appears which lists the number of topics, subtopics, and lines in the HELP file. The message is for informational purposes only.

3. Use your word processor to add or revise the text in the HELP.DAT file. See Editing Your HELP.DAT File below. Be sure to save your file with the changes and return to DOS.
4. At the MS-DOS prompt, invoke the HELP command. To make HELP reflect your changes, enter *C (for *create*), as follows:

```
A>HELP(Enter topic, ESC to exit)> *C
```

After a minute or two, the message

```
HELP.HLP created
```

appears. The next time you invoke the HELP command, you will see the changes you made.

Editing your HELP.DAT File

If you are adding a new topic or subtopic, go to the appropriate place in the text. The topics are listed in alphabetical order. You must type in the heading with the following format:

///ntopicname

where *///* designates a topic in the file, *n* is the topic level (1 is a main topic; 2 is a subtopic; 3 is a subtopic of a subtopic; etc.; up to eight levels of subtopics), and *topicname* is the name of the topic or subtopic.

Note: You must leave *two blank spaces* at the left margin before you begin to type in the topic heading.

A *topicname* can be any combination of 12 alphanumeric characters. Only topic names at level 1 appear in the list of HELP command main topics. You cannot leave a space within a topic name.

Under the topic heading, type in the information you want to include. What you type in is what you see when you select that topic from the HELP screen.

If you are adding or revising text of existing topics, just go to the desired topic and make your revisions. Remember to save your changes before exiting to MS-DOS if you are using a word processor.

JOIN

The JOIN command joins a disk drive to a specific pathname.

Type: External

JOIN has the following format:

```
JOIN d: [pathname] [/D]
```

where *d:* is the letter of the drive you want to join to *pathname*. This makes all of the directories on *d:* accessible by specifying *pathname*.

If the path does not exist, MS-DOS tries to make a directory with that path. The path must be empty. After you issue the JOIN command, drive letter *d:* becomes invalid, and if you try to use it, MS-DOS displays the error message, "Invalid drive specification."

The JOIN command removes the distinction that physical drives are separately addressable by drive letter. With JOIN, you can refer to all the directories on a specific drive by using a single pathname. If the pathname already existed before the JOIN command was issued, it will not be usable while JOIN is in effect.

You can join a drive only at the root. The following command is valid:

```
C>JOIN D: C:\MEMOS
```

but this command is invalid:

```
C>JOIN D: C:\MEMOS\JUNE
```

You cannot join a drive while it is the current drive. For example, you cannot specify:

```
A>JOIN A: C:\EXTRA
```

because drive A is the current drive. First you must change to drive C (or another drive), then you can issue this command.

To unjoin a drive from a pathname, use the following format:

```
JOIN d: /D
```

where *d:* is the joined drive and the /D switch turns off the JOIN command.

If you simply type:

```
JOIN
```

MS-DOS displays the current drives that are joined.

LABEL

The LABEL command creates, changes, or deletes the volume label on a disk device.

Type: External

LABEL has the following format:

```
LABEL d: [volume label]
```

where *d:* is a letter specifying the device for which to create, change, or delete the volume label. *volume label* is the new volume label.

Volume labels can be up to 11 characters long, including spaces. You cannot use the following characters in a volume label:

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

If you omit the volume label, LABEL will prompt you for it, showing the current one, if there is one.

If you want to delete the volume label, just press Return when prompted for the volume label. You will be prompted again:

```
Delete current volume label (Y/N)?
```

Enter Y to delete the volume label, or N to leave it unchanged.

NOTE: You cannot use the LABEL command over a network.

MKDIR

The MKDIR (or MD, for *make directory*) command creates a new directory.

Type: Internal

The MKDIR command has either of the following formats:

MKDIR *pathname*
MD *pathname*

where *pathname* can be one or more existing directory names separated by backslashes. The last name is the name of the new directory.

When you create directories with MKDIR, they always appear under your working directory unless you explicitly specify a different pathname.

The following example creates a subdirectory \USER in your root directory:

```
MKDIR \USER
```

The following example creates a directory named \JOE under \USER:

```
MKDIR \USER\JOE
```

MODE

The MODE command provides 11 options for controlling output devices. The functions provided by each option are:

- □ Option 1—controls output to the printer.
- □ Option 2—sets the display mode for the screen and the external monitor.
- □ Option 3—determines communications protocol for the serial port.
- □ Option 4—changes parallel printer output to a serial printer.
- □ Option 5—redirects the default printer output to another printer.
- □ Option 6—controls power to the serial port.
- □ Option 7—controls power to the optional internal modem, if installed, and controls the emulated Hayes Smartmodem™ switch settings.
- □ Option 8—determines the blink and shape of the cursor.
- □ Option 9—controls power to the internal diskette drive.
- □ Option 10—sets font size and reverse video.
- □ Option 11—determines which device letter ROM files are appended to.

Type: External

Each option is explained in detail in the sections that follow.

Option 1—Controlling the Printer

Option 1 of the MODE command determines the width and vertical spacing on the documents produced by your printer. Specify Option 1 as follows:

```
MODE LPTn[:] [80 | 132][,6 | 8][,P]
```

The parameters are described in the table below.

Parameter	Setting
LPTn:	Specify LPT1:, LPT2:, or LPT3: for the printer number to direct the output to one of three printers attached to your computer. Note the following: LPT1 always refers to the default printer (same as PRN). LPT2 or LPT3 refer to a printer plugged into an expansion unit. If you have more than one printer, and want to determine the applicable number for each one, print a few lines of text alternately using the LPT1, LPT2, and LPT3 settings.
80 or 132	Specifies the number of characters per line. The initial setting is 80, which remains in effect unless you change it with this command.
6 or 8	Specifies the number of vertical lines per inch. The initial setting is 6, which remains in effect unless you change it with this command.
P	Specify P when you wish to resend output after a time-out occurs. The system continuously reissues printing requests until either the printer becomes active or you press Fn-Break. You must reenter Option 1 of the MODE command without the P to disable this automatic retry action. Use this option with printers with slower baud rates, which sometimes create a time-out condition during certain operations—for example, when ejecting to the next page. The time-out causes a halt in the printing process; specifying P prevents such halts.

Examples The following are examples of Option 1 of the MODE command.

```
MODE LPT1:,8
```

This example changes the number of vertical lines to 8 per inch. The number of characters per line aren't affected—the comma (,) specifies the current value of characters per line.

```
MODE LPT1:132,6
```

This example changes the number of characters per line to 132, and the number of vertical lines to 6 per inch.

Option 2—Setting Display Mode

Option 2 of the MODE command sets the display mode for your computer screen and the external monitor. Specify Option 2 as follows:

MODE n
or
MODE [n][,L | R][,T]

The Option 2 parameters are explained in the following sections.

n The *n* parameter specifies both the type of display attached to your computer and the number of characters per line as explained in the table below.

n setting	Result
40	40 characters per line appear on your computer screen and the external display.
80	80 characters per line appear on your computer screen and the external display.
BW40	40 characters per line appear on your computer screen and the external display. If the external display is a color monitor, the screen is in black and white.
BW80	Same as BW40, except 80 characters per line appear on the display.
CO40	40 characters per line appear on your computer screen and the external display. If the external display is a color monitor, the screen is in color.
CO80	Same as CO40, except 80 characters per line appear on the displays.
MONO	Specifies the monochrome display adapter, which is 80 characters per line.

L or R (Shift) This parameter affects only an external monitor. Specify either *L* (for left) or *R* (for right). When specified, the display shifts one character (for 40 characters per line) or two characters (for 80 characters per line) to either the left or right.

Use this option with the Test option (described next) to align an off-center display on an external monitor.

T (Test) This parameter affects only an external monitor; it requests a test pattern to be displayed after specifying a shift; you can align a display that is off-center using this facility.

You must specify an *L* or *R* shift parameter for this parameter to take effect. When you specify *T*, the display is shifted as specified and a test pattern is displayed; MS-DOS prompts you to indicate if the screen is aligned properly.

- □ If you enter *Y* (for yes), the command ends.
- □ If you enter *N* (for no), the screen shifts again and you are again prompted to indicate if the screen is aligned properly. This sequence is repeated until you specify *Y*.

Option 3—Serial Adapter Protocol

Option 3 of the MODE command specifies the protocols to be used when communicating with devices connected through either the COM1 or COM2 ports. Specify Option 3 as follows:

```
MODE COMn[:]baud[,parity[,databits[,stopbits[,P[,DTR + | -[,RTS + | -]]]]]]
```

The Option 3 parameters are explained in the table below.

Parameter	Setting
COMn:	Specify COM1: or COM2:, indicating the port number. This is a required parameter; you must always specify a port number. The following rule applies if you specify COM1: If the internal modem has been activated by Option 7 (MODE MODEM = ON), then the new protocol applies to the internal modem. Otherwise, the new protocol applies to the serial port; power to the serial port, if off, is turned on.
baud	Specify one of the following to indicate the baud rate: 110, 150, 300, 600, 1200, 2400, 4800, or 9600; if desired, specify only the first two characters of the baud rate (any other characters are ignored). For example *11* indicates a baud rate of 110, and *12* a rate of 1200. This is a required parameter; you must always specify a baud rate.
parity	Specify either N (none), O (odd), or E (even). This is an optional parameter; the default is E.
databits	Specify either 7 or 8. This is an optional parameter; the default is 7.
stopbits	Specify either 1 or 2. This is an optional parameter; the default is 2 if the baud rate is 110; otherwise the default is 1.
P	Specify P when a serial printer is connected, and you wish to resend output after a time-out occurs. The system continuously reissues printing requests until either the printer is active or you press Fn-Break. You must reenter Option 3 of the MODE command without the P to disable this automatic retry action.
DTR	Specify DTR + to enable DTR (Data Terminal Ready) for serial port handshaking. (Note that setting DTR is normally not required.)
RTS	Specify RTS + to enable RTS (Request To Send) serial port handshaking. (Note that setting RTS is normally not required.)

The following rules apply when accepting default values for parameters:

□ To accept all the default parameters, specify only the port number and baud rate. For example, the following sets the baud rate to 1200 in the COM1 port; the parity is E (even), databits are 7, and stopbits are 1.

```
MODE COM1:1200
```

The system then displays the following message:

```
COM1:1200,e,7,1,-
```

(The message shows the default values that are in effect; the hyphen (-) at the end of the line indicates that *P* was not specified.)

□ To accept selected default parameters, specify a comma instead of a value for *parity*, *databits*, or *stopbits*. For example, the following sets the *stopbits* to 2 in the COM1 port; the baud rate is 1200.

```
MODE COM1:1200,,,2
```

The system then displays the following message:

```
COM1:1200,e,7,2,-
```

(The message shows both the values you specified and the default values.)

Option 4—Changing Parallel Printer Output

Option 4 of the MODE command causes output normally printed on a parallel printer to be redirected to a printer connected through the COM1 or COM2 port.

NOTE: Issue Option 1 of the MODE command to direct output back to the parallel printer.

Specify Option 4 as follows:

MODE LPTn[:] = COMn

The Option 4 parameters are explained below.

Parameter	**Setting**
LPTn:	Specify either LPT1:, LPT2:, or LPT3: for the printer number.
COMn	Specify either COM1 or COM2 for the serial port number.

Option 5—Redirect Default Printer Output

Option 5 of the MODE command allows you to select the printer (LPT1:, LPT2:, or LPT3:) that will be used to print output directed to the PRN device.

Specify Option 5 as follows:

```
MODE PRN[:] = 1 | 2 | 3
```

Whenever you specify PRN as the output device, output will be sent to the printer selected here (LPT1:, LPT2:, or LPT3:). The initial setting is 1.

Option 6—Switch On Serial Port

Option 6 of the MODE command switches on the power to the internal serial port and assigns to it the symbolic device name COM1. Power to the port is off after start-up (booting).

The format of Option 6 is as follows:

```
MODE SERIAL = ON | OFF
```

To turn on the power to the serial port, specify Option 6 as follows:

```
MODE SERIAL=ON
```

The system then displays the following message:

```
COM1: status—serial power ON, modem power OFF
```

When you turn on the power to the serial port, the power to the internal modem, if on, is automatically turned off.

After executing MODE SERIAL = ON, the following protocol is in effect for serial communications:

Parameter	**Setting**
baud	2400
parity	N (none)
databits	8
stopbits	1

To change these settings, use Option 3 of the MODE command.

To turn off the power to the serial port, specify Option 6 as follows:

```
MODE SERIAL=OFF
```

The system then displays the following message:

```
COM1: status—serial power OFF, modem power OFF
```

Option 7—Switch On Internal Modem

Option 7 of the MODE command performs the following functions:

□ Turns on or off the power to the optional internal modem and assigns to it the symbolic device name COM1. Power to the port is off after start-up (booting).

□ Controls the Hayes Smartmodem switch settings emulated by the internal modem.

□ Allows you to turn the internal speaker on or off.

The format of Option 7 is as follows:

```
MODE MODEM = ON | OFF [,addressSwitch][,ON | OFF]
```

The first ON or OFF parameters respectively power on or power off the internal modem. The second ON or OFF parameters respectively turn on or turn off the internal speaker while the modem is dialing and waiting for a connection. The speaker always automatically turns off when a connection is established.

The Hayes™ Smartmodem address settings are represented in *addressSwitch*; these settings are determined by the MS-DOS communications application you are using. See the user documentation for the application to determine the correct settings of the switches.

In the MODE command, *addressSwitch* consists of eight characters of *U*'s and/or *D*'s (for *up* or *down*, meaning the switch is respectively "on" or "off").

The initial (default) setting is UDDDUUUD, meaning Switches 1, 5, 6, and 7 are up (on) and Switches 2, 3, 4, and 8 are down (off). These are the settings required by the CROSSTALK™ application.

The table below reflects the meaning of each switch setting.

Switch	Position[1]	Meaning
1	**Up**	Modem requires DTR active
	Down	Modem does not require DTR active
2	Up	Result codes sent as English words
	Down	Result codes sent as single digits
3	Up	No result codes are sent
	Down	Result codes are sent
4	Up	Echo characters when in command mode
	Down	No echo
5	**Up**	Auto answer on first ring
	Down	Don't answer
6	**Up**	Carrier detect real
	Down	Carrier detect forced
7	**Up**	Always up
8	Up	Disables command recognition
	Down	Enables command recognition

[1] Default settings are shown in bold.

To turn on the power to the internal modem, and to set the switches to the Hayes factory setting, specify Option 7 as follows:

```
MODE MODEM=ON,DUDUDDUD
```

The system then displays the following message:

```
COM1: status—modem power ON, serial power OFF.
```

The switches in the internal modem now indicate the following settings:

Switches 1, 3 ,5, 6, and 8 are down (off).
Switches 2,4, and 7 up (on).

These settings are canceled and the initial (default) settings are restored when you do one of the following:

- □ Issue the command MODE MODEM = OFF
- □ Restart MS-DOS.

When you turn on the power to the internal modem, the power to the serial port, if on, is automatically turned off.

After executing MODE MODEM = ON, the following protocol is in effect:

Parameter	**Setting**
baud	1200
parity	N (none)
databits	8
stopbits	1

To change these settings, use Option 3 of the MODE command.

To turn off the power to the internal modem, specify Option 7 as follows:

```
MODE MODEM=OFF
```

The system then displays the following message:

```
COM1: status—modem power OFF, serial power OFF.
```

Option 8—Setting Cursor Appearance

Option 8 of the MODE command determines how the cursor appears on your display.

Specify Option 8 as follows:

MODE CURSOR = [LINE | BLOCK],[BLINK | NOBLINK]

The parameters are described in the table below.

Parameter	Meaning
LINE or BLOCK	If you specify LINE (the initial setting), the cursor appears as a short, horizontal underline. If you specify BLOCK, the cursor appears as a small highlighted rectangle.
BLINK or NOBLINK	If you specify BLINK (the initial setting), the cursor flickers on and off; if you specify NOBLINK, the cursor retains a constant glow.

Examples The following are examples of Option 8 of the MODE command.

This example causes the current cursor shape to be steady and non-blinking. The shape is not changed.

```
MODE CURSOR=,NOBLINK
```

This example causes the cursor to appear as a blinking, solid rectangle.

```
MODE CURSOR=BLOCK,BLINK
```

Option 9—Powering Internal Diskette Drive

Option 9 of the MODE command powers on or off the internal diskette drive.

Specify Option 9 as follows:

```
MODE DISK=[ON | OFF]
```

The default setting is OFF. Specify ON only for those programs written to access the disk controller directly rather than through the ROM BIOS.

When DISK = OFF, the power is on only when a program accesses the internal diskette. When DISK = ON, the power to the internal diskette is always on.

Option 10—Setting Font and Reverse Video

Option 10 of the MODE command controls how characters appear on the screen.

Specify Option 10 as follows:

```
MODE FONT = [1 | 2],[RVON | RVOFF]
```

The parameters are explained in the table below.

Parameters	Meaning
1 or 2	The 1 and 2 options control the font size of the characters on your screen. Specify 1 (the initial setting) to make the characters appear "thick" or 2 to make the characters appear "thin."
RVON or RVOFF	RVOFF (the initial setting) turns reverse video off; dark characters with a light background appear on the display. RVON turns reverse video on; light characters with a dark background appear. This option affects only the computer screen; the screen of an external monitor isn't changed.

Option 11—Setting the ROM Device Letter

Option 11 of the MODE command sets the device letter that all ROM files, if available, are appended to.

Specify Option 11 as follows:

```
MODE ROM = A | B | OFF
```

The initial setting is A. If you select A or B, ROM files are appended to drives A or B, respectively. If you select OFF, ROM files are not appended to any drive, and thus are not available for use.

MORE

The MORE command sends output to the screen one screen at a time.

Type: External

MORE has the following format:

```
MORE
```

MORE is a filter that reads from standard input (such as a command from your keyboard) and displays one screen of information at a time. Then, the MORE command pauses and displays the —MORE— message at the bottom of your screen.

To display the next screen, press any key. This process continues until all the input data has been read.

The MORE command is useful for viewing a long file one screen at a time. If you type

```
TYPE MYFILES.TXT | MORE
```

(Press Fn-Shift-| to make the | character)

MS-DOS displays the file MYFILES.TXT (on the default drive) one screen at a time; you scroll through each successive screen by pressing any key.

PATH

The PATH command sets a command search path.

Type: Internal

PATH has the following format:

PATH [pathname[;pathname] . . .]

This command allows you to tell MS-DOS which directories should be searched for external commands after MS-DOS searches your current directory. The default value is no path.

To tell MS-DOS to search your \BIN \USER \JOE directory for external commands, type:

```
PATH \BIN\USER\JOE
```

MS-DOS then searches the current directory, and if the file is not found, searches the \BIN \USER \JOE directory for external commands until you set another path or re-start MS-DOS.

You can tell MS-DOS to search more than one path by specifying several pathnames separated by semicolons. For example,

```
PATH \BIN\USER\JOE;\BIN\USER\SUE;\BIN\DEV
```

tells MS-DOS to search the directories specified by the above pathnames to find external commands. MS-DOS searches the pathnames in the order specified in the PATH command.

The command PATH with no options displays the current path.
If you specify

```
PATH;
```

MS-DOS sets the NUL path, meaning that only the current directory is searched for external commands.

The PATH command is frequently used in an AUTOEXEC.BAT file. See Chapter 8, Batch Files, for details.

PCSLAVE The PCSLAVE command operates along with the PCMASTER.SYS device driver to permit communication between a GRiDCase and an IBM PC, PC/XT, or PC/AT through their serial ports. One computer acts as the master station, controlling the transfer operations, and the other acts as a slave station. Issuing PCSLAVE enables the slave station. See chapter 11 for details on using this command for file transfer.

Both computers must be connected with the appropriate serial-to-serial cable. See Chapter 11 for details on the cable.

Type: External

The PCSLAVE command has the following format:

```
PCSLAVE [1 | 2]
```

If the slave station is a GRiDCase, power to the serial port, if off, is turned on and remains on until you press Ctrl-C.

Issuing PCSLAVE or PCSLAVE 1 indicates that transfer takes place through the COM1 serial port of the computer.

Issuing PCSLAVE 2 indicates that transfer takes place through the COM2 serial port (of an attached expansion unit or on the IBM PC, PC/XT, or PC/AT).

After issuing the command, the following messages appear:

```
Slave station ready
Enter commands at master station
Enter Ctrl-C upon completion
```

NOTE: You must issue the PCSLAVE command at the slave station *before* starting up the master station. This allows MS-DOS at the master station to recognize and assign letters to the storage devices available at the slave station.

PRINT

The PRINT command prints a text file on a printer while you are processing other MS-DOS commands (usually called "background printing").

Type: External

PRINT has the following format:

PRINT [[/Q:value][filespec] [/D:device] [/B:size] [/T] [/C] [/P]] . . .

Use the PRINT command only when a printer is attached to your computer. The following switches are provided with this command:

/D:device	Specifies the print device. If not specified, the default device is PRN.

The following switches are allowed only the first time you run the PRINT command after starting MS-DOS.

/Q:value	Specifies the number of files allowed in the print queue. The minimum value is 4, the maximum is 10. If not specified, the default value is 10. This switch must be specified before *filespec*.
/B:size	Sets the size in bytes of the internal buffer. Increasing the value of *size* speeds up the PRINT command.
/T (terminate)	Deletes all files in the print queue (those waiting to be printed). A message to this effect is printed.
/C (cancel)	Turns on cancel mode. The parameter *filespec* and all following *filespecs* are removed from the print queue until you type a /P switch.
/P (print)	Turns on print mode. The preceding *filespec* and all following filespecs are added to the print queue until you issue a /C switch.

PRINT with no options displays the contents of the print queue on your screen without affecting the queue.

Examples

The following are examples of the PRINT command.

```
PRINT /T
```

empties the print queue.

```
PRINT A:TEMP
.TST/C A:TEMP2.TST A:TEMP3.TST
```

removes the three files indicated from the print queue.

```
PRINT TEMP
.TST /C TEMP2.TST /P TEMP3.TST
```

removes TEMP1.TST from the queue, and adds TEMP2.TST and TEMP3.TST to it.

PROMPT

The PROMPT command changes the MS-DOS command prompt.

Type: Internal

PROMPT has the following format:

PROMPT [prompt-text]

This command allows you to change the MS-DOS system prompt (for example, A>). If no text is typed, the prompt is set to the default prompt, which is the default drive designation. You can set the prompt to some other character or characters by using the dollar sign followed by a special character as shown in the following table.

Character	**Resulting Prompt**
$$	$
$t	Current time
$d	Current date
$p	Current directory of the default drive
$v	Version number
$n	device letter of the default drive
$g	> character
$l	< character
$b	| character
$_	CR LF sequence
$s	Space (leading only)
$h	Backspace
$e	ASCII code X'1B' (escape)
blank	device letter of the default drive followed by >

Examples

The following are examples of the PROMPT command:

```
PROMPT $n$g
```

sets the default drive letter prompt.

```
PROMPT Time = $t$_Date = $d
```

sets a two-line prompt that prints:

```
Time = (current time)
Date = (current date)
```

If you have installed an ANSI escape sequence driver in your CONFIG.SYS file, then you can specify escape sequences in your prompt. For example:

```
PROMPT $e[7m$n:$e[m
```

displays the prompt in reverse video then returns to normal video.

Specifying PROMPT with no parameters returns to the default prompt

RECOVER The RECOVER command recovers a file or an entire disk containing bad sectors.

Type: External

RECOVER has the following format:

```
RECOVER filespec | d:
```

If a sector on a disk is bad, you can recover either the file containing that sector (without the bad sector) or the entire disk (if the bad sector was in the directory).

To recover a particular file, type:

```
RECOVER filespec
```

This causes MS-DOS to read the file sector by sector and to skip the bad sector(s). When MS-DOS finds the bad sector(s), they are marked bad and not used again.

To recover a disk, type:

```
RECOVER d:
```

where d: is the letter of the drive containing the disk to be recovered.

CAUTION: Be careful when you use the RECOVER command. When you use it to recover a disk, it renames all of the files using the format FILE0001.REC, FILE0002.REC, etc. It is recommended that you backup the files before you attempt to recover a disk.

NOTE: You cannot use the RECOVER command over a network.

REN

The RENAME (or REN) command lets you change the name of a file or, when using wildcards, a group of files.

Type: Internal

The RENAME command has either of the following formats:

```
RENAME filespec filename
REN filespec filename
```

where *filespec* identifies the name of the file you want to rename, and *filename* specifies the new filename.

You can include wildcards in *filespec* and *filename*. For example, the following command changes the names of all files with the .LST extension to similar names with the .PRN extension:

```
REN *.LST *.PRN
```

In the next example, RENAME renames the file ABODE in the diskette in drive A to ADOBE:

```
REN A:?B?D? ADOBE
```

REPLACE

The REPLACE command duplicates a source file or group of source files to one or more target files with the same names.

Type: External

The REPLACE command has the following format:

```
REPLACE filespec1 [filespec2] [/A] [/D] [/P] [/R] [/S] [/W]
```

where *filespec1* is the source file or directory and *filespec2* is the target directory.

The files in the target directory are replaced with files in the source directory that have the same name. You may use wildcards in source filenames.

If you specify the /A switch, REPLACE adds files to the target directory that exist in the source directory but **not** in the target directory (i.e. only new files are added).

The following switches are provided with REPLACE:

/A Causes REPLACE to add only new files to the target directory, instead of replacing existing ones. You cannot use this switch with the /D or /S switches.

/D Replaces files in the target directory only if the source files are newer than the corresponding target files. You cannot use this switch with the /A switch.

/P Causes REPLACE to prompt you before replacing or adding each file.

/R Replaces read-only files as well as other files. If you don't specify this switch, any attempt to replace a read-only file causes an error and stops the replace process.

/S Causes REPLACE to search all subdirectories of the target directory while replacing files. It never searches subdirectories in the source path. You cannot use this switch with the /A switch.

/W Waits for you to press any key before replacing files; otherwise, replacing or adding files begins immediately.

As files are replaced or added, the filenames are displayed on the screen; at the conclusion of the replace operation, the number of files replaced or added is reported.

As files are replaced or added, the filenames are displayed on the screen; at the conclusion of the replace operation, the number of files replaced or added is reported.

You cannot use the REPLACE command to update hidden files or system files.

Examples

The following are examples of the REPLACE command:

Replacing files:

Suppose your hard disk, drive C, contains several files of client names and phone numbers. To replace these files with the latest version of this file that exists on the disk in drive A, you would enter:

```
REPLACE A:\PHONES.CLI C:\ /S
```

This command replaces every file on drive C that is named PHONES.CLI with the file PHONES.CLI from the root directory on drive A.

Adding files:

Suppose you want to add some new printer device drivers to a directory called C:\MSTOOLS, which already contains several printer driver files for a word processor. To do this, you would enter:

```
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.

RESTORE

The RESTORE command restores one or more files that you backed up with the BACKUP command.

Type: External

The RESTORE command has the following format:

RESTORE d: [d:] [pathname] [/S] [/P] [/A:mm-dd-yy] [/B:bb-dd-yy]
[/E:hh:mm] [/L:hh:mm] [/M] [/N]

where the first *d:* is the disk drive where you backed up your files and the second *d:* and *pathname* specify the drive and files you want to restore. The files that you want to restore must have been placed on the backup disk with the BACKUP command.

If you don't specify a directory, the backup files are restored to the current directory. If you don't specify a filename or extension, all files from the directory that were backed up are restored.

You can use a wildcard character in the filename to restore all files that match the specified filename. For example,

```
C>RESTORE A: C:*.COM
```

restores all files from the backup disk with the .COM extension that were backed up from the current directory. When restoring files from multiple floppy diskettes to a hard disk, after MS-DOS restores all the files on the inserted diskette that match the specified filename, MS-DOS prompts you to insert the next backup diskette so it can continue the restore procedure.

The following switches are provided with RESTORE:

/S	Restores files at all levels of subdirectories, as well as the files in the specified directory.
/P	Makes RESTORE prompt you before it restores any hidden or read-only files. Some files are marked read-only when they are created.
/A:mm-dd-yy	Restores only those files that were last modified on or after the given date.
/B:mm-dd-yy	Restores only those files that were last modified on or before the given date.
/E:hh:mm	Restores only those files that were last modified at or earlier than the given time.
/L:hh:mm	Restores only those files that were last modified at or later than the given time.

/M Restores only those files that have been modified since the last backup.

/N Restores only those files that no longer exist on the destination disk.

If you are restoring files that were backed up using a previous version of MS-DOS, you should use the /P switch. This switch prompts you before restoring system files. You should type "N" when prompted to avoid destroying the newer version of MS-DOS on your disk.

NOTE: This RESTORE command and the one supplied by IBM are compatible except for the /A, /B, /E, /L, /M, and /N switches.

Examples

The following example restores all files on the backup diskettes to the hard disk (assuming C: is the hard disk drive):

```
C>RESTORE A: C:\ /S
```

The following example restores two files from a backup diskette to the hard disk drive:

```
C>RESTORE A: C:\EMPLOYEE\CURRENT
C>RESTORE A: C:\EMPLOYEE\CURRENT\FEMALE
```

Using the RESTORE Command with Batch Files

When you complete the RESTORE command within a batch file, you can test for the following exit codes with the IF command.

Code	Status
0	Normal completion
1	No files were found to restore
2	Some files not restored due to sharing conflicts
3	Terminate by user
4	Terminated due to error

RMDIR

The RMDIR (or RD, for *remove directory*) command deletes a subdirectory.

Type: Internal

The RMDIR command has either of the following formats:

RMDIR [pathname]
RD [pathname]

You must delete all files in the directory before using the RMDIR command. Only . and . . entries can remain in the directory to be deleted.

For example, suppose you want to remove EAST from the following directory structure:

```
\SALES\EXPENSES\EAST
```

You might use the following procedure to remove EAST.

1. Make EAST the current directory with the change directory command:

   ```
   CD \SALES\EXPENSES\EAST
   ```

2. Next, display all the files in EAST with the DIR (for *directory*) command:

   ```
   DIR
   ```

3. Copy all the files in EAST to a backup device (a diskette drive in the example below):

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

4. Delete all the files in EAST:

   ```
   DEL *.*
   ```

   (Respond *Y* to the "Are you sure?" prompt.)

5. Go to the parent directory, EXPENSES.

   ```
   CD ..
   ```

6. Delete EAST with RD as follows:

   ```
   RD EAST
   ```

SCANCASE This command shows the status of input/output devices attached to your GRiDCase and lists the factory-installed hardware options. The command is useful in the following instances:

- □ To find out if your GRiDCase has the built-in options your applications require.
- □ To provide information to GRiD personnel responsible for analyzing possible malfunctions in your system.

Type: External

SCANCASE has the following format:

```
SCANCASE
```

The following summarizes the information displayed by SCANCASE:

ROM BIOS Date. (Read Only Memory Basic Input Output System). ROM BIOS is the basic component of MS-DOS installed in the GRiDCase when it is manufactured. The date discloses when the BIOS code for your GRiDCase was written.

8087. States whether the internal numeric processor is installed or not installed in your GRiDCase. The processor is designed for faster calculations in graph, worksheet, and other arithmetic applications.

Available RAM. (Random Access Memory) The number of bytes of main memory in your GRiDCase.

Internal Floppy. Gives the device letter assigned to the internal diskette drive. The notation *active* indicates that the device is operating properly; any other message indicates a possible malfunction.

Keyboard ROM. Gives the version number of the ROM that stores the code controlling GRiDCase keyboard functions.

Active External Bus Device(s). Lists the devices attached to the external bus connector of the GRiDCase, and the date the version of the code for the internal ROM of each GRiD device was written; the notation *attached peripheral* appears for each non-GRiD device attached.

Parallel (Centronics) Interface. States whether a printer or other device is attached to the parallel connector.

Modem. States whether a GRiD-installed internal modem is present in your GRiDCase and gives the checksum of the modem PROM.

NOTE: Do not use the SCANCASE command after you have powered on the internal modem with the MODE MODEM = ON command. If you do use SCANCASE after the modem is powered on, you will need to re-start (boot) your computer before the modem will operate properly again.

SET

The SET command sets one string value equivalent to another string.

Type: Internal

SET has the following format:

SET [string = string]

This command is meaningful only if you want to set values that are used by programs you have written. The SET command inserts the string you enter into a part of memory reserved for "environment" strings. If the first *string* already exists in the environment, it is replaced by the new string. For example, SET TTY = VT52 sets your TTY value to VT52 until you change it with another SET command.

If you enter the SET command with only the first *string*, the associated string name is removed from the environment.

The SET command can also be used in batch processing. In this way, you can define your replaceable parameters with names instead of numbers. If your batch file contains the statement "LINK %FILE%," you can set the name that MS-DOS uses for that variable with the SET command.

The command

```
SET FILE=DOMORE
```

replaces the %FILE% parameter with the filename DOMORE. Therefore, you don't 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 be ended by a percent sign (%).

If you type SET with no arguments, MS-DOS displays the current setting of SET.

SHARE

The SHARE command installs file sharing and locking when operating a network.

Type: External

SHARE has the following format:

```
SHARE [/F:space] [/L:locks]
```

The SHARE command is only used when networking is active. Include it in the AUTOEXEC.BAT file to install shared files.

The following switches are provided with SHARE:

/F:space	Allocates file space (in bytes) for the area MS-DOS uses to record filesharing information. Each file that is open needs the length of the full filename plus 11 bytes (the average pathname is 20 bytes). The default value for the /F switch is 2048 bytes.
/L:locks	Allocates the number of locks you want to allow. The default value for the /L switch is 20 locks.

Once you have used the SHARE command in an MS-DOS session, all read and write requests are checked by MS-DOS.

The following example loads file sharing and uses the default values for the /F and /L switches:

```
SHARE
```

SORT

The SORT command arranges data from a file you specify in numerical and alphabetical order (from 0 to 9, A to Z), and then writes it to the screen or to a file you specify. In sorting alphabetical characters, the SORT command ignores upper- and lowercase alphabetical characters.

Type: External

SORT has the following format:

```
SORT [filespec] [/R] [/ + n]
```

SORT can be used, for example, to alphabetize a file by a certain column.

The following switches are provided with SORT:

/R Reverse the sort; that is, sort from Z to A, 9 to 0.

/ + n *n* is a column number specifying the start of the sorting field. If you don't specify this switch, SORT begins sorting from column 1.

Examples

This command reads the file UNSORT.TXT, sorts the file in reverse alphabetical order, and then writes the output to a file named SORT.TXT:

```
SORT /R <UNSORT.TXT >SORT.TXT
```

The following command sorts the output of the directory starting with column 14 (which contains the file size); the sorted output is then displayed on the screen. (See Table 6-1 in Chapter 6 for details on directing output.) Thus, the result of this command is a directory sorted by file size:

```
DIR |SORT /+14
```

The command

```
DIR |SORT /+14 | MORE
```

achieves the same result as the previous example, except that MORE displays the directory one screen at a time; each subsequent screen appears by pressing any key.

SUBST

The SUBST command substitutes a string for a pathname.

Type: External

SUBST has the following format:

SUBST d: [pathname] [/D]

where *d:* is the letter of the virtual drive that you want substituted for *pathname*.

Whenever you use the new virtual drive letter, MS-DOS replaces it with the specified pathname. This is useful for replacing commonly used pathnames with a single drive letter for convenient referencing.

Note that only recognized drive letters can be used for *d:* . If you want to use a drive letter that is not normally available on your system, you must add sufficient drives through use of the LASTDRIVE command in the CONFIG.SYS file. See LASTDRIVE in Chapter 5 for details.

If you use SUBST alone, the names of any virtual drives in effect are displayed.

Use the /D switch to delete a drive associated with a pathname.

NOTE: You cannot use the SUBST command over a network.

Examples

The command

```
SUBST F: B: \USR \FRED \FORMS
```

creates a virtual drive F for the pathname B: \USER \FRED \FORMS. Now, instead of typing this full pathname, you can get to this directory by simply typing F:.

The command

```
SUBST F: /D
```

deletes the virtual drive F.

SYS

The SYS command transfers the two MS-DOS system files from the default device to any other storage medium you specify—hard disk, diskette, or other storage device.

Type: External

SYS has the following format:

```
SYS d:
```

The two system files (IO.SYS and MSDOS.SYS) are both hidden files that don't appear in reports produced by the DIR (for *directory*) command.

When replacing an old version of MS-DOS, the new version may require more space than the one already on the target device. Try to transfer the new version. If you get the message "No room for system on destination diskette," back up any essential files on the target device and reinitialize it with the FORMAT command. Note that initializing a device destroys any existing files on it.

NOTE: You cannot use the SYS command over a network.

TIME

The TIME command displays the current time as recorded by the internal clock of your computer and lets you change it, if you wish. The time you set determines the time you see in a file directory that tells when the file was created or last modified.

Type: Internal

TIME has the following format:

TIME [<hh>[:<mm>][:<ss>][.<cc>]]

Entering the command TIME displays the following information:

```
Current time is hh:mm:ss.cc
Enter new time:_
```

Press Return if you don't want to change the time shown, or type in a new time in the format hh:mm; *hh* represents the hour and can be any number from 00 through 23; *mm* represents the minute and can be any number from 00 through 59. For example, you enter 8:20 a.m. and 8:20 p.m. in the following formats:

```
8:20
20:20
```

Or you can bypass the prompt by entering the command and correct time together in either of these formats:

```
TIME 8:20
TIME 20:20
```

NOTE: When setting the time, you don't have to enter *ss* (seconds) or *cc* (hundredths of seconds). You must always, however, separate the hour and minutes by a colon.

TREE

The TREE command displays a list of directories and the subdirectories defined within each directory. If you so specify, TREE also lists the names of the files in each directory and subdirectory.

Type: External

TREE has the following format:

```
TREE [d:][/F]
```

where *d:* is a device letter indicating the drive that contains the directories; if you don't specify *d:* , MS-DOS lists the directories on the default drive.

Specifying the /F switch causes the names of the files in each directory and sub-directory to be displayed.

You can print a list of directories and filenames by redirecting the output to a printer with the following command:

```
TREE B: /F > PRN
```

TYPE

The TYPE command displays the contents of a file on the screen.

Type: Internal

TYPE has the following format:

```
TYPE [d:] filename
```

After entering the TYPE command and pressing Return, the system scrolls the data in *filename* on the display. To temporarily halt the scrolling, press Ctrl-S; to resume scrolling, press any key.

When you display a binary file (for example, executable program code normally contained in files with the extensions .COM, .EXE, and .OBJ), a variety of special characters appear along with the alphabetic, numeric, and special characters ordinarily seen.

You can use the TYPE command in conjunction with the MORE filter when you display large files on the screen.

VER

The VER (for *version*) command displays the release number of MS-DOS currently controlling the operations of your computer. You may need this information when experiencing operating difficulties and want to make sure that you are operating with the appropriate MS-DOS version.

Type: Internal

VER has the following format:

```
VER
```

VERIFY

The VERIFY command controls the verify switch when writing to disk.

Type: Internal

VERIFY has the following format:

```
VERIFY [ON | OFF]
```

This command has the same effect as the /V switch in the COPY command. If you want to verify that all files are written correctly to disk, you can use the VERIFY command to tell MS-DOS to verify that your files are intact (no bad sectors, for example). MS-DOS performs a VERIFY each time you write data to a disk. You receive an error message only if MS-DOS was unable to successfully write your data to disk.

VERIFY ON remains in effect until you change it in a program (by a SET VERIFY system call), or until you issue a VERIFY OFF command to MS-DOS.

If you want to know what the current setting of VERIFY is, enter VERIFY with no options.

VOL

The VOL (for *volume*) command displays a disk volume label, if one exists.

Type: Internal

VOL has the following format:

```
VOL [d:]
```

This command displays the volume label of the disk in drive *d:* If no drive is specified, MS-DOS displays the volume label of the disk in the default drive.

XCOPY

The XCOPY command copies files and directories, including lower level directories if they exist.

Type: External

XCOPY has the following format:

```
XCOPY filespec1 [filespec2] [/A] [/D:mm-dd-yy] [/E] [/M] [/P] [/S] [/V] [/W]
```

where *filespec1* is the source file or directory, and *filespec2* is the target file or directory. If you omit *filespec2*, XCOPY assumes you want to copy the files to the default directory.

If you omit the path from either *filespec*, XCOPY uses the default directory. If you omit the filename from either *filespec*, XCOPY uses the default *.*, meaning all files in the specified directory.

The following switches are provided with XCOPY:

/A	Copies all source files that have their archive bit set. It does not modify the archive bit of the source file. See the ATTRIB command for information on how to set the archive attribute of a file.
/D:mm-dd-yy	Copies all source files that were modified on or after the date specifed by *mm-dd-yy*.
/E	Copies any subdirectories, even if they are empty. You can only use this switch if you also specify the /S switch.
/M	Copies all source files that have their archive bit set, and turns off the archive bit in the source file. See the ATTRIB command for information on how to set the archive attribute of a file.
/P	Prompts you with "(Y/N?)," allowing you to confirm whether or not you want to create each target file.
/S	Copies directories and lower level subdirectories, unless they are empty. If you omit this switch, XCOPY works within a single directory. (If you specify the /E switch along with the /S switch, even empty subdirectories are copied.)
/V	Verifies each file as it is written to the target, to make sure that the target files are identical to the source files.

/W	Waits before beginning file copying. The message "Press any key when ready to start copying files" is displayed. Press any key to start, or press Ctrl-C to abort XCOPY.

NOTE: XCOPY should be used when copying an entire disk that has files in subdirectories to another disk that has a different format. The DISKCOPY command cannot be used since it requires the source and target disks to have the same format.

Using MCOPY Instead of XCOPY

The XCOPY command may prompt you to specify whether the target is a file or a directory. If you do not want to receive this prompt, you can use the MCOPY command. To create the MCOPY command, enter

```
COPY /B XCOPY.EXE MCOPY.EXE
```

Now you can use the MCOPY command the same way you use the XCOPY command, but MCOPY automatically determines whether the target is a file or directory. MCOPY uses the following rules for copying files:

□ If the source is a directory, the target is a directory.

□ If the source includes multiple files, the target is a directory.

□ If you append a backslash (\) to the end of the target name, the target is a directory. For example, the command

```
MCOPY PAYROLL A: \WORKERS\
```

creates the directory A: \WORKERS if it doesn't already exist, and copies the file PAYROLL to it.

Examples

The following command:

```
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.

The following command:

```
MCOPY C: \WP A:\WPBACKUP\/M /S
```

copies the files and subdirectories under the \WP directory on drive C to the directory \WPBACKUP on drive A. Only those source files with their archive bits set are copied, and once they are copied, the archive bits are turned off. Any empty subdirectories are not copied.

The following command:

```
XCOPY A: \MEMOS B: \MEMOS /D:3-1-86 /S
```

copies the files and subdirectories under the \MEMOS directory on drive A to the directory \MEMOS on drive B. Only those source files modified on or after 3-1-86 are copied. Any empty subdirectories are not copied.

Using the XCOPY Command with Batch Files

When you complete an XCOPY command within a batch file, you can test for the following exit codes with the IF command.

Code	Status
0	Normal completion
1	No file found to copy
2	Terminated by user (Ctrl-C pressed)
4	Initialization error—not enough memory, invalid drive or command syntax, file not found, or path not found
5	INT 24 error—user aborted from INT 24 error reading or writing disk. Common INT 24 errors include disk read or write errors, write-protect errors, and drive not ready errors.

Chapter 8: Batch Files

A batch file contains DOS commands and has the extension .BAT. You can create a batch file with the COPY command (described below), with EDLIN, or with any other text editor. When you type in the name of the file (without the extension) and press RETURN, MS-DOS executes the commands it contains.

You can assign any valid filename to a batch file; you can also assign the filename AUTOEXEC.BAT, which MS-DOS executes automatically at start-up, bypassing the time and date prompts normally displayed on the screen.

MS-DOS treats batch files as external commands; that is, a batch command must always reside in the current directory or a path directory when you execute it.

Comments and General Rules

When you create batch files, remember that only the filename should be entered to execute the batch file. Do not enter the filename extension, since MS-DOS automatically executes the commands in the file *filename*. BAT.

If you press Ctrl-C while in batch mode, this prompt appears:

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

If you press Y, the remainder of the commands in the batch file are ignored and the system prompt appears.

If you press N, only the current command ends and batch processing continues with the next command in the file.

- □ If you remove the disk containing a batch file being executed, MS-DOS prompts you to insert it again before the next command can be read.
- □ The last command in a batch file can have the name of another batch file. This allows you to call one batch file from another when the first is finished.
- □ You can use the following commands in batch files (they are described later in this chapter):

  ECHO
  FOR
  GOTO
  IF
  PAUSE
  REM
  SHIFT

AUTOEXEC.BAT File

At start-up, MS-DOS searches for the AUTOEXEC.BAT file; if found, MS-DOS automatically executes the commands in it and suppresses the normal date and time prompts.

How To Create a Batch File

The following example shows how to create a batch processing file. Assigning the name AUTOEXEC.BAT causes the file to execute automatically at start-up; assigning any other valid filename requires that you enter that filename for the commands to execute.

The purpose of the following AUTOEXEC.BAT file is to set some system configuration options and then run the program WordStar from a subdirectory.

1. Type:

   ```
   COPY CON \AUTOEXEC.BAT
   ```

   This statement tells MS-DOS to copy the information from the keyboard into the AUTOEXEC.BAT file. Remember that you must create the AUTOEXEC.BAT file in the root directory of your MS-DOS disk.

2. Now enter the following statements:

   ```
   MODE LPT1:=COM1
   MODE SERIAL=ON
   MODE COM1:1200
   ```

   These statements go into the AUTOEXEC.BAT file. The first statement redirects printer output to the serial port where a serial printer is attached. The second statement turns on power to the serial port. The last statement configures the serial output port.

3. Enter the following statement:

   ```
   MODE ROM=B
   ```

   This statement appends all ROM files to drive B.

4. Enter the following statements:

   ```
   CD \WORDPROC
   WS
   MODE SERIAL=OFF
   ```

These statements change the current directory to \WORDPROC, run the program WordStar, and then turn off power to the serial port when WordStar is finished.

5. Press Fn-6 (or Ctrl-Z); then press Return to halt the insertion of characters into the AUTOEXEC.BAT file.

   The statements entered above now run automatically whenever you start MS-DOS.

You can enter any MS-DOS command or series of commands in the AUTOEXEC.BAT file.

Creating Batch Files with Replaceable Parameters

You can create a batch (.BAT) file with dummy (replaceable) parameters, which you specify when you actually execute the command.

The following example shows how to create a batch file using replaceable parameters.

Enter the following statements:

```
A>COPY CON MYFILE.BAT
COPY %1.MAC %2.MAC
TYPE %2.PRN
TYPE %0.BAT
```

Now, press Fn-6 (or Ctrl-Z) and Return. MS-DOS responds with this message:

```
1 File(s) copied
A>_
```

The file MYFILE.BAT, which consists of three commands, now resides on the disk in the default drive.

The dummy parameters %1 and %2 are replaced sequentially by the parameters you supply when you execute the file. The dummy parameter %0 is always replaced by the device letter, if specified, and the filename of the batch file (for example, MYFILE).

Note the following when specifying replaceable parameters in batch files:

- □ You can specify up to ten dummy parameters (%0 to %9). Refer to the SHIFT command in this chapter if you wish to specify more than ten parameters.
- □ If you use the percent sign as part of a filename within a batch file, you must type it twice. For example, to specify the file ABC%.EXE, you must type it as ABC%%.EXE in the batch file.

Executing with Replaceable Parameters To execute the batch file MYFILE.BAT and specify the parameters to replace the dummy parameters, enter the batch filename (without its extension) followed by the parameters you want MS-DOS to substitute for %1, %2, etc.

To execute the MYFILE batch process, type:

```
MYFILE A:PROG1 B:PROG2
```

MYFILE is substituted for %0, A:PROG1 for %1, and B:PROG2 for %2.

The result is the same as if you had typed each of the commands in MYFILE with their parameters, as follows:

```
COPY A:PROG1.MAC B:PROG2.MAC
TYPE B:PROG2.PRN
TYPE MYFILE.BAT
```

Remember that the dummy parameter %0 is always replaced by the drive designator (if specified) and the filename of the batch file.

A description of the commands available for use within batch files follows in the remaining part of the chapter.

ECHO

The ECHO command permits you to control the display of commands and remarks within a batch file as the file is executed.

Type: Internal

The ECHO command has the following format:

ECHO [ON | OFF | message]

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 echo back on.

If ON or OFF are not specified, the current setting is displayed.

ECHO <message> is useful if ECHO is off and you are using a batch file. By typing ECHO and a message in your batch file, you can display messages on the screen.

FOR

The FOR command allows multiple execution of commands within a batch file.

Type: Internal

The FOR command has either of the following formats:

FOR %%c IN set DO command (for batch processing)
FOR %c IN set DO command (for interactive processing)

where *c* can be any character except 0,1,2,3, . . . ,9 to avoid confusion with the %0-%9 batch parameters.

set is (list of items)

The %%c variable is set sequentially to each member of *set*, and then *command* is evaluated. If a member of *set* is an expression involving * and/or ?, then the variable is set to each matching pattern from the disk. In this case, only one such *item* may be in the set, and any *item* besides the first is ignored.

Examples

The following are examples of the FOR command:

```
FOR %%f IN ( *.ASM ) DO MASM %%f;
FOR %%f IN (REPORT MEMO ADDRESS) DO DEL %%f
```

The first example binds the variable named %f to files ending with .ASM in the working directory. It then executes the command:

```
MASM filename
```

where *filename* could be: BIGFILE.ASM, SORTER.ASM, and LIST.ASM.

The second example binds the variable %f to the files named REPORT, MEMO, and ADDRESS. It then deletes each of these files.

The '%%' is needed so that after batch parameter (%0-%9) processing is done, there is one '%' left. If only '%f' were there, the batch parameter processor would see the '%', look at 'f', decide that '%f' is an error (bad parameter reference) and throw out the '%f', so that the command FOR would never see it. If the FOR is not in a batch file, then only one '%' should be used.

GOTO

The GOTO command permits branching within a batch file as it executes.

Type: Internal

The GOTO command has the following format:

GOTO label

GOTO causes commands to be taken from the batch file beginning with the line after the *label* definition. If no label has been defined, the current batch file will terminate.

Example

The following is an example of the GOTO command:

```
:loop
REM looping ...
GOTO loop
```

will produce an infinite sequence of messages:

```
REM looping....
```

Starting a line in a batch file with ':' causes the line to be ignored by batch processing. The characters following GOTO define a label, but this procedure may also be used to put in comment lines.

IF

The IF command permits the conditional execution of DOS commands within a batch file.

Type: Internal

The IF command has the following format:

IF condition command

The parameter *condition* is one of the following:

ERRORLEVEL number	True if and only if the previous program executed by COMMAND had an exit code of *number* or higher.
string1 = = string2	True if and only if *string1* and *string2* are identical after parameter substitution. Strings may not have embedded separators.
EXIST filename	True if and only if *filename* exists.
NOT condition	True if and only if *condition* is false.

The IF statement allows conditional execution of commands. When the *condition* is true, then the *command* is executed. Otherwise, the *command* is ignored.

Examples

The following are examples of the IF command:

```
IF NOT EXIST TAX ECHO Can't find file
IF NOT ERRORLEVEL 3 LINK %1,,;
```

PAUSE

The PAUSE command suspends execution of the batch file.

Type: Internal

The PAUSE command has the following format:

PAUSE [comment]

During the execution of a batch file, you may need to change disks or perform some other action. PAUSE suspends execution until you press any key, except Ctrl-C.

When the command processor encounters PAUSE, it prints:

```
Strike a key when ready ...
```

If you press Ctrl-C, another prompt will be displayed:

```
Abort batch job (Y/N)?
```

If you type Y in response to this prompt, execution of the remainder of the batch command file will be aborted and control will be returned to the operating system command level. Therefore, PAUSE can be used to break a batch file into pieces, allowing you to end the batch command file at an intermediate point.

The comment is optional and may be entered on the same line as PAUSE. You may also want to prompt the user of the batch file with some meaningful message when the batch file pauses. For example, you may want to change disks in one of the drives. An optional prompt message may be given in such cases. The comment prompt will be displayed before the "Strike a key" message.

REM (REMARK)

The REM command lets you insert comments among the other statements in a batch file.

Type: Internal

The Remark command has the following format:

REM [comment]

The only separators allowed in the comment are the space, tab, and comma.

Example

The following is an example of the Remark command:

```
REM This file does a directory.
PAUSE Insert new disk in drive B:
DIR B:
```

The above sequence, when executed, causes the following text to appear on the screen:

```
REM This file does a directory.
Insert new disk in drive B:
Strike any key when ready ...
DIR B:
```

SHIFT

The SHIFT command allows access to more than 10 replaceable parameters in batch file processing.

Type: Internal

The Shift command has the following format:

```
SHIFT
```

Usually, command files are limited to handling 10 parameters, %0 through %9. To allow access to more than 10 parameters, use SHIFT to change the command line parameters. For example, if

%0 = "foo"
%1 = "bar"
%2 = "name"
%3 . . . %9 are empty

then a SHIFT will result in the following:

%0 = "bar"
%1 = "name"
%2 . . . %9 are empty

If there are more than 10 parameters given on a command line, those that appear after the tenth (%9) are shifted one at a time into %9 by successive shifts.

Note that there is no backward shift. Once SHIFT is executed, the %0 parameter that existed before the shift cannot be recovered.

Chapter 9: The Line Editor (EDLIN)

The Line Editor (EDLIN) program performs the following functions:

□ Creates new source files and saves them.

□ Updates existing files and saves both the updated and original files.

□ Deletes, edits, inserts, and displays lines.

□ Searches for, deletes, or replaces 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 line numbers following the inserted text advance automatically by the number of lines being inserted. 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 Enter and Exit EDLIN

To start EDLIN, type:

```
EDLIN filespec
```

If you are creating a new file, the *filespec* should be the name of the file you wish to create. If EDLIN does not find this file on a drive, EDLIN creates a new file with the name you specify.

To exit EDLIN, enter either "Exit" or "E" next to the EDLIN prompt (discussed below), which is an asterisk (*).

How To Enter Text

The following message and prompt are displayed after you enter EDLIN:

```
New file *_
```

Notice that the prompt for EDLIN is an asterisk (*). You can now enter one of the EDLIN commands described later in the chapter. You can now proceed as follows:

□ To enter a command, type in the command (or initial as indicated in the descriptions later in the chapter) and press Return.

For example, to enter insert mode so that you can enter text, type in I (for INSERT) and press Return.

Start entering text immediately to the right of the line number and an asterisk appears on the screen. Press Return for the next sequential line number when desired.

□ To leave insert mode, press Ctrl-Z. Enter the desired command immediately to the right of the asterisk and press Return.

If you want to edit an existing file, *filespec* 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.

Editing Files Too Large for Main Memory (RAM)

If the file is too large to be loaded into memory, EDLIN loads lines until memory is three-quarters full, 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; then EDLIN can load the unedited lines from disk into memory. Refer to the WRITE and APPEND commands in this chapter for the procedure.

Exiting EDLIN

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 described later in this chapter.) The original file is renamed with an extension of .BAK, and the new file has the filename and extension you specified in the EDLIN command.

The original .BAK file isn't erased until the end of the editing session, or until disk space is needed by the editor (EDLIN). Do not try to edit a file with a filename 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 RENAME command described in Chapter 7), then start EDLIN and specify the new *filespec* .

Special Editing Keys

Special editing keys are available for you to edit text files. Note that these keys also work at the MS-DOS command level. Table 9-1 summarizes these editing keys and their functions.

Table 9-1. *MS-DOS Editing Keys*

Key[*]	**Function**
F1	Press F1 to copy one character from the template (the template refers to the last command or line of text you entered) and display the character. Press and hold F1 to copy the entire line. The → key provides the same function.
F2	Copies all characters from the template to the new line, up to the character you enter after pressing F2.
F3	Copies all remaining characters in the template to the screen.
F4	Skips over the characters in the template, up to the character you specify after pressing F4.
F5	Accepts an edited line for continued editing. The current line becomes the template, but it is not executed.
Ins	Pressing Ins allows you to insert characters within a line as you copy it from the template. Pressing Ins again halts the insert mode. You can then press F1 to continue copying or Return to confirm the command.
Del	Skips over one character in the template. The cursor doesn't move.
Esc	Cancels the line currently being displayed and, if you press Return, erases the template.

[*] The keys listed here represent function keys on your computer. Your computer may use different keystrokes than those listed here to implement the same function. For example, on some computers, you must press Fn-F1 to get F1, or Fn-Ins to get Ins. Refer to the owner's manual for your computer for specific details.

EDLIN Commands—General Rules

EDLIN commands perform editing functions on lines of text. The following list contains information you should read before you use EDLIN commands.

1. Pathnames are acceptable as options to commands. For example, typing

   ```
   EDLIN \BIN\USER\JOE\TEXT.TXT
   ```

   allows you to edit the TEXT.TXT file in the subdirectory \JOE.
2. 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. For example, typing

   ```
   -10,+10L
   ```

   lists 10 lines before the current line, the current line, and 10 lines after the current line.
3. Multiple commands can be issued on one command line. When you issue a command to edit a single line using a line number, a semicolon must separate commands on the line. Otherwise, one command may follow another without any special separators. In the case of a SEARCH or REPLACE command, the *string* may be ended by pressing Fn-6 (Ctrl-Z) instead of Return.

   For example, the following command line edits line 15 and then displays lines 10 through 20 on the screen.

   ```
   15;-5,+5L
   ```

   The command line in the next example searches for "This string" and then displays 5 lines before and 5 lines after the line containing the matched string. If the search fails, then the displayed lines are those line numbers relative to the current line.

   ```
   SThis string<Ctrl-Z>-5,+5L
   ```

4. 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.

5. You can insert a control character (such as Ctrl-C) into text by using the quote character Ctrl-V before it while in insert mode. Ctrl-V tells MS-DOS to recognize the next **capital** letter typed as a control character (lowercase letters are ignored). It is also possible to 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>foo
   ```

   replaces all occurrences of Ctrl-Z in a file by "foo".

   ```
   r<Ctrl-V>C<Ctrl-Z>bar
   ```

   replaces all occurrences of Ctrl-C by "bar".

   To insert Ctrl-V into the text, enter Ctrl-V and then "V."

6. The Ctrl-Z character ordinarily tells EDLIN, "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 switch to tell EDLIN to ignore any Ctrl-Z characters in the file and to show you the entire file. For example, when you start EDLIN, type:

   ```
   EDLIN filespec /B
   ```

   The EDLIN commands are summarized below. They are also described in further detail after the description of command options.

EDLIN Command Options

Several EDLIN commands accept one or more options. The effect of a command option varies, depending with which command it is used. Table 9-2 describes each option.

Table 9–2. *EDLIN Command Options*

Option	Description
line	Indicates a line number that you type. Line numbers must be separated by a comma or a space from other line numbers and other options. *line* may be specified one of three ways: **Number.** Any number less than 65534. If a number larger than the largest existing line number is specified, then *line* means the line after the last line number. **Period (.).** If a period is specified for line, then line means the current line number. The current line is the last line edited, and is not necessarily the last line displayed. The current line 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,* this has the same effect as specifying a number larger than the last line number. Return. A Return entered without any of the line specifiers listed above directs EDLIN to use a default value appropriate to the command.
?	The question mark option directs EDLIN to ask you if the correct string has been found. The question mark is used only with the REPLACE and SEARCH commands. Before continuing, EDLIN waits for either a Y or Return for a "yes" response, or for any other key for a "no" response.
string	*string* represents text to be found, to be replaced, or to replace other text. The *string* option is used only with the SEARCH and REPLACE commands. Each string must be ended by a Ctrl-Z or Return (see the REPLACE command for details). No spaces should be left between strings or between a string and its command letter, unless you want those spaces to be part of the string.

EDLIN Command Summary

Below is a summary of the commands described in the sections that follow.

Command	Purpose
line #	Displays line to be edited
A	Appends lines
C	Copies lines
D	Deletes lines
E	Ends editing
I	Inserts lines
L	Lists lines
M	Moves lines
P	Pages lines
Q	Quits editing
R	Replaces lines
S	Searches text
T	Transfers text
W	Writes lines

APPEND The APPEND command adds the specified number of lines from disk to the file being edited in memory. The lines are added at the end of lines that are currently in memory.

The APPEND command has the following format:

```
[n]A
```

This command is meaningful only if the file being edited is too large to fit into memory. When you start EDLIN, as many lines as possible are read into memory for editing.

To edit the remainder of the file that won't fit into memory, lines that have already been edited must be written to disk. Then you can load unedited lines from disk into memory with the APPEND command. Refer to the WRITE command in this chapter for information on how to write edited lines to disk.

Note the following when using the APPEND command

□ If you do not specify the number of lines to APPEND, lines are appended to those already in memory until available memory is three-quarters full. No action is taken if available memory is already three-quarters full.

□ The message "End of input file" is displayed when the APPEND command has read the last line of the file into memory.

COPY

The COPY command copies a range of lines and inserts them before a specified line number. The lines can be copied as many times as you want by using the *count* option.

The COPY command has the following format:

[line],[line],line[,count]C

where the first *line* specifies the beginning of the range of lines you want to copy and the second *line* specifies the end of the range you want to copy. The third *line* parameter specifies the line **before** which you want the copied text to appear. *Count* specifies how many copies of the selected lines you want to make.

If you do not specify a number in *count*, EDLIN copies the lines one time. If the first or the second *line* is omitted, the default is the current line. The file is renumbered automatically after the copy.

The line numbers must not overlap or an "Entry error" message appears. For example,

```
3,20,15C
```

would result in an error message.

Examples

Assume that the following file exists and is ready to edit:

```
 1: This is a sample file
 2: used to illustrate copying lines.
 3: See what happens when you use
 4: the COPY command
 5: (the C command)
 6: to copy text in your file.
```

You can copy this entire block of text by issuing the following command:

```
1,6,7C
```

This will copy lines 1 through 6, duplicating them one time, beginning on line 7. The result is:

```
 1: This is a sample file
 2: used to illustrate 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 illustrate copying lines.
 9: See what happens when you use
10: the COPY command
11: (the C command)
12: to copy text in your file.
```

If you want to place the text within other text, the third *line* option should specify the line before which you want the copied text to appear. For example, assume that you want to copy lines and insert them within the following file:

```
 1: This is a sample file
 2: used to illustrate 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 illustrate 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.
```

DELETE

The DELETE command deletes a specified range of lines in a file.

The DELETE command has the following format:

[line][,line]D

where the first *line* specifies the beginning of the range of lines to delete, and the second *line* specifies the end of the range of lines to delete.

If the first *line* is omitted, that option will default to the current line (the line with the asterisk next to the line number). If the second *line* is omitted, then just the first *line* is deleted. When lines have been deleted, the line immediately after the deleted section becomes the current line and has the same line number as 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, 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 text
 7:*in your file.
```

To DELETE a single line, type:

```
6D
```

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 text
 7: in your file.
```

To DELETE a range of lines beginning with the current line, type:

```
,6D
```

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.

EDIT

The EDIT command edits a line of text.

The EDIT command has the following format:

[line]

When a line number is typed, EDLIN displays the line number and text; then, on the line below, EDLIN reprints the line number. The line is now ready for editing. You may use any of the EDLIN editing commands to edit the line. The existing text of the line serves as the template until Return is pressed.

If no line number is typed (that is, if you press the Return key), the line after the current line (marked with an asterisk (*)) is edited. If no changes to the current line are needed and the cursor is at the beginning or end of the line, press Return to accept the line as is.

Example

Assume that the following file exists and is ready to edit:

```
 1: This is a sample file.
 2: used to show
 3: the editing of line
 4:*four.
```

To edit line 4, type:

```
4
```

The contents of the line are displayed with a cursor below the line:

```
4:*four.
4:*_
```

Now type:	Result:
InS number	4: number_
F3	4: number four.
Return	*

END

The END command ends the editing session.

The END command has the following format:

```
E
```

This command saves the edited file on disk, renames the original input file *filename*.BAK, and then exits EDLIN. If the file was created during the editing session, no .BAK file is created.

The END command takes no options. Therefore, you cannot tell EDLIN on which drive to save the file. The drive you want to save the file on must be selected when the editing session is started. If the drive is not selected when EDLIN is started, the file is saved on the disk in the default drive. It is still possible to COPY the file to a different drive using the MS-DOS COPY command.

You must 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 aborted and the edited file lost, although part of the file might be written out to the disk.

Example

The following is an example of the END command:

```
E
```

After execution of the END command, the MS-DOS default drive prompt (for example, A>) is displayed.

INSERT

The INSERT command inserts text immediately before the specified line.

The INSERT command has the following format:

```
[line]I
```

where *line* is a line number in the file being edited.

If you are creating a new file, the I command must be given before text can be typed (inserted). Text begins with line number 1. Successive line numbers appear automatically each time Return is pressed.

EDLIN remains in insert mode until you press Ctrl-C. When the INSERT is completed and insert mode is exited, the line immediately following the inserted lines becomes the current line. All line numbers following the inserted section are incremented by the number of lines inserted.

If *line* is not specified, the default is the current line number and the lines are inserted immediately before the current line. If *line* is any number larger than the last line number, or if a pound sign (#) is specified as *line*, 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 illustrate dynamic line numbers
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7:*in your file.
```

To insert text before line 7, type

```
7I
```

The result is:

```
7:
```

Now, type the new text for line 7:

```
7: and renumber lines
```

Then to end the insertion, press F6 (or Ctrl-Z) when line 8 appears:

```
8: F6 (or Ctrl-Z)
```

Now type L to list the file. The result is:

```
 1: This is a sample file
 2: used to illustrate dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7:  and renumber lines
 8:*in your file.
```

To insert lines immediately at the current line, type:

```
I
```

The result is:

```
 8:
```

Now, insert the following text and insert a Ctrl-Z on the next line by pressing F6 (or Ctrl-Z):

```
 8: so they are consecutive
 9: F6 (or Ctrl-Z)
```

Now to list the file and see the result, type L:

The result is:

```
 1: This is a sample file
 2: used to illustrate dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit 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:

```
10I
```

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 65533.
```

End the insertion by pressing Ctrl-Z on line 14. The new lines will 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 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 65533.
```

LIST

The LIST command lists a range of lines, including the two lines specified.

The LIST command has the following format:

[line1][,line2]L

where *line1* is the first line to be displayed and *line2* is the final line.

Default values are provided if either one or both of the options are omitted. If you omit the *line1,* as in:

,line2 L

the display starts 11 lines before the current line and ends with *line1.* The beginning comma is required to indicate the omitted first option.

NOTE: If *line2* is more than 11 lines before the current line, the display is the same as if you omitted both options.

If you omit *line2* , as in:

line1 L

23 lines are displayed, starting with *line* .

If you enter *L* , omitting *line1* , and *line2* , 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 less than 11 lines before the current line, more than 11 lines after the current line are displayed to make a total of 23 lines.

Examples

Assume that the following file exists and is ready to edit:

```
  1: This is a sample file
  2: used to illustrate 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 text
 27: in your file.
```

To list lines 2 through 5 without reference to the current line, type

```
2,5L
```

The result is:

```
 2: used to illustrate 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

```
15,26L
```

The result is:

```
15:*The current line contains an
     -
     -
     -
26: to edit text
27
```

To list a range of 23 lines centered around the current line, type only

```
L
```

The result is:

```
 4: DELETE and INSERT
 5: (the D and I commands)
     -
     -
     -
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 text.
```

MOVE

The MOVE command moves a range of text to the line specified.

The MOVE command has the following format:

[line1],line2,line3M

Use the MOVE command to move a block of text (beginning with *line1* and ending with *line2*) to another location in the file (*line3*). The lines are renumbered according to the direction of the move.

Example

The following is an example of the MOVE command:

```
,+25,100M
```

moves the text from the current line plus 25 lines to line 100.

If the line numbers overlap, EDLIN displays an "Entry error" message.

To move lines 20 to 30 to line 100,type:

```
20,30,100M
```

PAGE

The PAGE command pages through a file 23 lines at a time from *line1* to *line2* .

The PAGE command has the following format:

```
[line1][,line2]P
```

If omitted, *line1* defaults to the current line plus one. If *line2* is omitted, 23 lines are listed. The new current line becomes the last line displayed and is marked with an asterisk.

QUIT

The QUIT command quits the editing session, does not save any editing changes, and exits to the MS-DOS operating system.

The QUIT command has the following format:

```
Q
```

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 in this chapter for information about the .BAK file.

Type N if you want to continue the editing session.

NOTE: When started, EDLIN 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 Abort edit (Y/N)? message, your previous backup copy is erased.

Example

The following is an example of the QUIT command:

```
Q Abort edit (Y/N)?Y
A >
```

REPLACE The REPLACE command replaces all occurrences of a string of text in the specified range with a different string of text or blanks.

The REPLACE command has the following format:

```
[line1][,line2][?]Rstring1<Ctrl-Z>string2
```

where *line1* is the first line number and *line2* is the final line number of a block of text to be searched for *string1*; if *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*, then the line is displayed once for each occurrence. When all occurrences of *string1* in the specified range are replaced by *string2*, the REPLACE command terminates and the asterisk prompt reappears.

If a second string is to be given as a replacement, then *string1* must be separated from *string2* with a Ctrl-Z character. *String2* must be ended by pressing Return.

If *string1* or *string2* is omitted, then REPLACE takes the old *string1* or *string2* as its value. If there is no old *string1*, i.e., this is the first replace done, then the replacement process is terminated immediately. If *string2* is omitted, then *string1* may be ended by pressing Return.

If *line1* is omitted in the range argument (as in *,line2*) then *line1* default to the line after the current line. If *line2* is omitted, then *line2* defaults to #. Therefore, this is the same as *line,#*. Remember that # indicates the line after the last line of the file.

If *string1* is ended by pressing Ctrl-Z and there is no *string2*, *string2* is taken as an empty string and becomes the new replace string. For example,

Enter:	**Then press:**
Rstring1	Ctrl-Z or Return

deletes occurrences of *string1*, but the sequence

Enter:	**Then press:**
Rstring1	Return
R	Return

replaces *string1* by the old *string2* and the old *string1* with the old *string2*, respectively. Note that "old" here refers to a previous string specified either in a SEARCH or a REPLACE command.

If the question mark (?) option is given, 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 Return key, then *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, EDLIN displays the asterisk prompt.

If you press any key besides Y or Return after the O.K.? prompt, the *string1* is left as it was in the line, and REPLACE goes to the next occurrence of *string1*. If *string1* occurs more than once in a line, each occurrence of *string1* is replaced individually, and the O.K.? prompt is displayed before each replacement. In this way, only the desired instances of *string1* are 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 illustrate dynamic line numbers.
 3: See what happens when you use
 4: DELETE and INSERT
 5: (the D and I commands)
 6: to edit 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 65533.
```

To replace all occurrences of *string1* "and" with *string2* "or" in a specified range, type:

Enter:	**Then press:**
`2,12 RandCtrl-Zor`	Return

The result is:

```
 4: DELETE or INSERT
 5: (the D or I commors)
 8: The INSERT commor can place new lines
```

Note that in the above replacement, some unwanted substitutions have occurred. To avoid these and to confirm each replacement, the same original file can be used with a slightly different command.

In the next example, to replace only certain occurrences of the first *string1* with the second *string2*, type:

Enter:	**Then press:**
`2? Rand`	Ctrl-Z or Return

The result is:

```
 4: DELETE or INSERT
O.K.? Y
 5: (The D or I commands)
O.K.? Y
 5: (The D or I commors)
O.K.? N
 8: The INSERT commor can place new lines
O.K.? N
* _
```

Now, type the LIST command (L) to see the result of all these changes:

```
L

    -
    -
 4: DELETE or INSERT
 5: (The D or I commands)
    -
 8: The INSERT command can place new lines
    -
    -
```

SEARCH The SEARCH command searches the specified range of lines for a specified string of text.

The SEARCH command has the following format:

[line1][,line2][?]SstringReturn

string must be ended by pressing Return. The first line that matches *string* is displayed and becomes the current line. If the question mark option is not specified, the SEARCH command terminates when a match is found. If no line contains a match for *string*, the message "Not found" is displayed.

If the question mark option (?) is included in the command, EDLIN displays the first line with a matching string; it then prompts you with the message O.K.?. If you press either the Y or Return, the line becomes the current line and the search terminates. If you press any other key, the search continues until another match is found, or until all lines have been searched (and the "Not found" message is displayed).

If *line1* is omitted, it defaults to the line after the current line. If *line2* is omitted, it defaults to # (line after last line of file), which is the same as *line* ,# S*string* .

If *string* is omitted, SEARCH takes the old string if there is one. (Note that "old" here refers to a string specified in a previous SEARCH or REPLACE command.) If there is not an old string (i.e., no previous SEARCH or REPLACE has been done), the command terminates immediately.

Examples

Assume that the following file exists and is ready for editing:

```
 1: This is a sample file
 2: used to illustrate dynamic line numbers.
 3: See what happens when you use
 4: DELETE and INSERT
 5: (the D and I commands)
 6: to edit 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 65533.
```

To search for the first occurrence of the string "and",

Enter:	**Then press:**
`2,12 Sand`	Return

The following line is displayed:

```
4: DELETE and INSERT
```

To get the "and" in line 5, modify the SEARCH command as follows:

Press:	**Screen display:**
F4,	
F3	`,12 Sand`
Return	

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 Return), the search continues, so type N here:

```
O.K.? N
```

Continue:

```
 5: (the D and I commands)
O.K.?
```

Now press Y to terminate the search:

```
O.K.? Y
* _
```

TRANSFER The TRANSFER command inserts (merges) the contents of *filename* into the file currently being edited at *line* . If *line* is omitted, then the current line is used.

The TRANSFER command has the following format:

```
[line]Tfilename
```

This command is useful if you want to put the contents of a file into another file or into the text you are typing. The transferred text is inserted at the line number specified by *line* and the lines are renumbered.

WRITE

The WRITE command 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.

The WRITE command has the following format:

```
[n]W
```

Where *n* is the number of lines you want to write to disk. This command is meaningful 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-quarters full.

To edit the remainder of your file, you must write edited lines in memory to disk. Then you can load additional unedited lines from disk into memory by using the APPEND command.

NOTE: If you do not specify the number of lines, lines are written until memory is three-quarters full. No action is taken if available memory is already more than three-quarters full. All lines are renumbered, so that the first remaining line becomes line number 1.

Chapter 10: File Comparison Utility

The File Comparison (FC) utility program compares the contents of two files. The differences between the two files can be output to the screen or to a third file. The files you compare can be either text or source files (files containing source statements of a programming language), or binary files (files output by the MACRO-86 assembler, the LINK utility program, or by a Microsoft high-level language compiler).

The comparisons are made in one of two ways: on a line-by-line or a byte-by-byte basis. The line-by-line comparison isolates blocks of lines that are different between the two files and prints those blocks of lines. The byte-by-byte comparison displays the bytes that are different between the two files.

Limitations on Source Comparisons

FC uses a large amount of memory as buffer (storage) space to hold the source files. If the source files are larger than available memory, FC will compare what can be loaded into the buffer space. If no match is found in the portions of the files in the buffer space, FC will display only the message:

```
Resynch failed. Files are too different.
```

For binary files larger than available memory, FC compares both files completely, overlaying the portion in memory with the next portion from disk. All differences are output in the same manner as those files that fit completely in memory.

NOTE: At least 68K of free memory is required for FC to perform a file comparison.

How To Use the File Comparison Utility

The syntax of FC is as follows:

FC [/A] [/B] [/C] [/L] [/LB nnn] [/N] [/T] [/W] [/nnnn] pathname1 pathname2

FC matches the first file (*pathname1*) against the second (*pathname2*) and reports any differences between them. For example:

```
FC B:\FOO\BAR\FILE1.TXT \BAR\FILE2.TXT
```

FC takes FILE1.TXT in the \FOO\BAR directory of disk drive B: and compares it with FILE2.TXT in the \BAR directory. Since no drive is specified for *pathname2*, FC assumes that the \BAR directory is on the disk in the default drive. FC bases the comparison on the switch settings you specify as described in the next section.

Switches

The following switches are provided with the FC utility program:

/A Abbreviates the output of an ASCII comparison. Instead of displaying all of the lines that are different, only the lines that begin and end each set of differences are displayed.

/B Forces a binary comparison of both files. The two files are compared byte-to-byte, with no attempt to resynchronize after a mismatch. The mismatches are printed as follows:

```
xxxxxxxx   yy   zz
```

(where xxxxxxxx is the relative address of the pair of bytes from the beginning of the file). Addresses start at 00000000; yy and zz are the mismatched bytes from file1 and file2, respectively. If one of the files contains less data than the other, then a message is printed out. For example, if file1 ends before file2, then FC displays:

fc: *filename1* longer than *filename2*

This switch is the default when you compare .EXE, .COM, .SYS, .OBJ, .LIB, or .BIN files.

/C causes the matching process to ignore the case of letters. All letters in the files are considered uppercase letters. For example: (underscore represents a tab or space)

Much_MORE_data_IS_NOT_FOUND

will match

much_more_data_is_not_found

If both the /W and /C options are specified, then FC will compress tabs and spaces and ignore case. For example, ___DATA_was_found____

will match:

data_was_found

/L Compares the files in ASCII mode. This switch is the default when you compare files that do not have the following extensions: .EXE, .COM, .SYS, .OBJ, .LIB, or .BIN.

/LB nnn Sets the internal line buffer to *nnn* lines. The default length of the internal buffer is 100 lines. Files that have more than this number of consecutive, differing lines will abort the comparison.

/N Displays the line numbers on an ASCII comparison.

/T Does not expand tabs to spaces. The default is to treat tabs as spaces to 8-column positions.

/W Causes FC to compress tabs and spaces during the comparison. Thus, multiple contiguous tabs or spaces in any line will be considered as a single space. Note that although FC compresses tabs and spaces, it does not ignore them. The two exceptions are beginning and ending tabs or spaces in a line, which are ignored. For example (note that an underscore represents a tab or space),

___More__data_to_be_found____

will match with:

More_data_to_be_found

and with:

________More________data_to_be_____found________

but will not match with:

____Moredata_to_be_found

/nnnn Specifies the number of lines required to match for the files to be considered as matching again after a difference has been found. If this switch is not specified, it defaults to 2.

Difference Reporting

The FC utility program reports the differences between the two files you specify by displaying the first filename, followed by the first line to match in both files, followed by the lines that differ between the files, followed by the next line to match in both files. FC then displays the name of the second file followed by the first line to match in both files, followed by the lines that are different, followed by the next line that matches. For example,

```
    . . .
    . . .
***** filename1
1st line to match file2 in file1
differences
next line to match file2 in file1

***** filename2
1st line to match file1 in file2
differences
next line to match file1 in file2
    . . .
    . . .
```

FC will continue to list each difference.

The default for the number of lines to match between the files is 2. (If you want to change this default, specify the number of lines with the /nnnn switch.)

If there are too many differences (involving too many lines), the program will simply report that the files are different and stop.

If no matches are found after the first difference is found, FC will display:

```
Resynch failed. Files are too different.
```

and will return to the MS-DOS prompt (for example, A>).

If the files compare without any differences, FC displays the following message:

```
fc: no differences encountered
```

Redirecting Output to a File

The differences and matches between the two files you specify will be displayed on your screen unless you redirect the output to a file. This is accomplished in the same way as MS-DOS command redirection (refer to Chapter 6, "Input and Output").

To compare File1 and File2 and then send the FC output to DIFFER.TXT, type:

```
FC File1 File2 >DIFFER.TXT
```

The differences and matches between File1 and File2 will be put into DIFFER.TXT on the default drive.

Examples

The examples below show how to use the FC utility program to compare two files (Example 1), to print differences using the same two source files (Example 2), and to force a binary comparison and display the differences (Example 3).

Example 1

Assume these two ASCII files are on disk:

ALPHA.ASM	BETA.ASM
A	A
B	B
C	C
D	G
E	H
F	I
G	J
H	1
I	2
M	P
N	Q
O	R
P	S
Q	T
R	U
S	V
T	4
U	5
V	W

```
W                   X
X                   Y
Y                   Z
Z
```

To compare the two files and display the differences on the screen, type:

```
FC ALPHA.ASM BETA.ASM
```

FC compares ALPHA.ASM with BETA.ASM and displays the differences on the screen. All defaults remain intact. (The defaults are: perform an ASCII comparison with an internal buffer length of 100 lines, treat tabs as spaces to 8-column positions, do not ignore case, do not display line numbers, and do not compress tabs and spaces.)

The output will appear as follows on the screen:

```
***** ALPHA.ASM
C
D
E
F
G
***** BETA.ASM
C
G
*****

***** ALPHA.ASM
I
M
N
O
P
***** BETA.ASM
I
J
1
2
P
*****

***** ALPHA.ASM
V
W
```

```
***** BETA.ASM
V
4
5
W
```

Example 2

You can print the differences on the printer using the same two source files. In this example, four successive lines must be the same to constitute a match.

Type:

```
FC /4 ALPHA.ASM BETA.ASM >PRN
```

The following output will appear on the printer:

```
***** ALPHA.ASM
C
D
E
F
G
H
I
M
N
O
P
***** BETA.ASM
C
G
H
I
J
1
2
P
*****

***** ALPHA.ASM
V
W
***** BETA.ASM
```

```
V
4
5
W
```

Example 3

This example forces a binary comparison and then displays the differences on the screen using the same two source files as were used in the previous examples.

Type:

```
FC /B ALPHA.ASM BETA.ASM
```

The /B switch in this example forces binary comparison. This switch and any others must be typed before the filenames in the FC command line. The following display should appear:

```
00000009        44        47
0000000C        45        48
0000000F        46        49
00000012        47        4A
00000015        48        31
00000018        49        32
0000001B        4D        50
0000001E        4E        51
00000021        4F        52
00000024        50        53
00000027        51        54
0000002A        52        55
0000002D        53        56
00000030        54        34
00000033        55        35
00000036        56        57
00000039        57        58
0000003C        58        59
0000003F        59        5A
fc: alpha.asm longer than beta.asm
```

Chapter 11: The PC File Transfer Utilities

This chapter describes the PC File Transfer utilities (PCMASTER.SYS and PCSLAVE.EXE) that transfer files from storage devices attached to an IBM PC, PC/XT, or PC/AT to storage devices attached to a GRiDCase, and vice versa. The programs run concurrently on a GRiDCase and on an IBM PC, PC/XT, or PC/AT which are connected by a cable attached to their serial ports.

The primary function of the PC File Transfer utilities is to transfer application programs and data files from an IBM 5 ¼ -inch diskette or a hard disk to a GRiDCase 3 ½ -inch diskette or hard disk.

The chapter covers the following topics:

□ Required hardware and software

□ Operational overview

□ How to run the utilities

□ Messages and troubleshooting

□ How device letters are assigned

Serial Cable Requirements

You need the GRiD Model 6201 PCMASTER/PCSLAVE Serial Cable. Plug one end of the cable into the serial port of the GRiDCase and the other end into the serial port of the IBM PC or PC/XT.

If you already have a serial-to-serial cable and a null modem adapter (you can purchase one from an electronics store), the GRiD 6201 cable may not be required. Simply plug one end of your serial cable into the serial port of one computer, and plug the other end into the null modem adapter, and the adapter into the port of the second computer.

CAUTION: There is not a standard null modem adapter, so all null modem adapters may not work. Before you buy or use a null modem adapter, make sure the in and out pins are connected to each other as shown in Table 11-1.

Table 11-1. *Null Modem Adapter Requirements for PC File Transfer*

Pin In	Pin Out
1	1
2	3
3	2
5	20
6	20
7	7
8	20
20	5,6,8

If you are using the standard serial port on an IBM PC/AT, you will need a 9-pin to 25-pin adapter cable, included with your PC/AT or available from your computer dealer.

Required Files

After start-up (boot), you designate one computer as the slave station using the PCSLAVE command; the other computer becomes the master station after you start it up; the master station controls the file exchange.

The files required at each station are listed below.

Master Station

The PCMASTER.SYS file must be on the MS-DOS diskette at the master station.

You must also create a CONFIG.SYS file at the master station; it requires the following statement:

```
DEVICE = PCMASTER.SYS [1 | 2]
```

Slave Station

The PCSLAVE.EXE file must be available on an active storage device.

NOTE: PCSLAVE.EXE and PCMASTER.SYS are shipped with the GRiD MS-DOS system files on both 3 ½ -inch and 5 ¼ -inch diskettes. Use the 3 ½ -inch diskette on the GRiDCase internal drive and the 5 ¼ -inch diskette on the IBM PC, PC/XT, or PC/AT drive.

Overview of Operations

You operate the utilities from the master station using the standard MS-DOS commands. When the GRiDCase is the master station, you can specify devices at both the master and slave stations in all of the commands, except the following:

CHKDSK
DISKCOMP
DISKCOPY
FDISK
FORMAT
LABEL
RECOVER
SUBST
SYS

You can execute the above commands sucessfully only on devices at the GRiDCase master station.

How To Run PC File Transfer

The next procedure describes how to transfer files from diskettes on an IBM PC or PC/XT to a 3 ½ -inch diskette on the GRiDCase.

NOTE: The computer system that is running the most current (highest numbered) version of MS-DOS must be the master station.

To Run PC File Transfer

1. Connect the GRiD Model 6201 PC PCMASTER/PCSLAVE cable (or your own null modem cable) to the serial ports of each computer.
2. Insert system diskettes into the drives at the slave and master stations.

   In Steps 3 and 4 below you designate either the GRiDCase or the IBM PC or PC/XT as the master station.
3. At the slave station:
   - □ Start up (boot) MS-DOS.

- □ If the PCSLAVE.EXE file isn't already on your system diskette, insert a diskette containing that file into a drive.
- □ Enter one of the following commands to activate the slave station and to indicate that the connecting cable is plugged into the COM1 port:

```
C>PCSLAVE
```

or

```
C>PCSLAVE 1
```

- □ If the connecting cable is plugged into the COM2 port, enter the following command to activate the slave station:

```
C>PCSLAVE 2
```

When PCSLAVE is ready, a display appears with the following messages in the center of the screen:

```
Slave station ready
Enter commands at master station
```

At the bottom of the screen, you'll see:

```
Enter Ctrl-C upon completion.
```

NOTE: It may take several seconds for these messages to appear. Don't go to the next step until you see these messages.

4. At the master station:

- □ Start up (boot) MS-DOS.

  NOTE: Always start up the master station *after* start-up of the slave station (Step 3). This allows the master station to accurately assign letters to the storage devices attached to the slave station.
- □ Rename the CONFIG.SYS file (if present), or copy it to a file with a different name.
- □ Create a new CONFIG.SYS containing the following statement (or add the statement to an existing CONFIG.SYS file):

```
DEVICE = PCMASTER.SYS [1 | 2]
```

where *1* or *2* refers to the serial port (COM1 or COM2) you are using at the master station. (See Chapter 5 for detailed instructions on creating a CONFIG.SYS file if you don't already know how to create one.)

□ Restart (boot) MS-DOS by turning the power off, then on. Because PCMASTER is installed in the CONFIG.SYS file, power to the serial port is automatically turned on and it is configured for PCMASTER communication (9600 baud, 8 databits, 1 stopbit, no parity). There is no need to turn on the serial port power with the MODE command.

□ Enter an acceptable MS-DOS command (COPY, DIR, DEL, or FC).

For example, assume that the master station is a GRiDCase and the only storage device available is an internal diskette drive. To copy all the files on the first diskette drive at the PC to the internal 3½-inch diskette drive on the GRiDCase, enter the following command.

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

The device letters you specify vary with the number of devices attached to each computer. See Assignment of Device Letters below for more information.

When the master station is a GRiDCase, the power to the serial port is turned on at start-up; the power remains on until you turn it off with the MODE command, as shown in Step 6.

5. At the slave station, press Ctrl-C after completing the file transfer.
6. At the master station, do the following:

□ If you don't intend to use PCMASTER/PCSLAVE again, you should remove the DEVICE = PCMASTER.SYS statement from the CONFIG.SYS file. The statement increases the amount of main memory (RAM) required by MS-DOS.

To remove the statement, either delete the file as follows:

```
A>DEL CONFIG.SYS
```

or, if a CONFIG.SYS file existed before Step 6, restore it by entering the following command:

```
A>COPY savename CONFIG.SYS
```

where *savename* is the filename of the original CONFIG.SYS file you saved in Step 4.

□ If the master station is a GRiDCase, turn off the power to the serial port by entering the following command:

```
MODE SERIAL=OFF
```

□ Restart MS-DOS.

Messages

This section describes the messages you can receive during a file transfer and the action you take to recover from the error conditions they describe.

Message: **Serial port specified does not exist on system**

What To Do: You issued a PCSLAVE command. Either the serial port you specified doesn't exist, or your computer does not have a serial port. To use the COM1 serial port, enter one of the following commands:

```
PCSLAVE
```

or

```
PCSLAVE 1
```

To use the COM2 serial port, enter the following command:

```
PCSLAVE 2
```

Message: **Insufficient memory**

What To Do: Check the CONFIG.SYS file for large allocations of memory for RAMDRIVE or other memory resident applications. Remove these allocations from the CONFIG.SYS file.

Message: **Initialization error**

What To Do: An error was encountered while trying to initialize the serial port. Make sure the serial cards are properly inserted in the PC or PC/XT.

Message: **PCSlave requires DOS version 2.10 or higher**

Message: The version of MS-DOS on the Slave station must be 2.10 or higher, otherwise, PCSlave will not work.

Message: **Data error reading drive *x****
Abort, retry, ignore?_

What To Do: First, make sure the null modem cable connecting the serial ports of the two computers is firmly inserted. Then, enter R (for *retry*).

Message: **Not ready error reading drive *x****
Abort, Retry, Ignore?_

What To Do: Make sure the diskette is inserted in the proper drive and the latch is secured.

Make sure the diskette is in the drive that corresponds to the device letter you specified in the command. See the section Assignment of Device Letters below to determine which device letters you should specify.

**x* is a letter indicating the storage device.

File Transfer Troubleshooting

If you have problems with the file transfer utilities, be sure to check the following items.

- □ Ensure that you have the most current versions of both PCMASTER.SYS and PCSLAVE.EXE. A newer version of one should never be used with an older version of the other.
- □ Ensure that the system running the highest numbered version of MS-DOS or PC-DOS is the master station.
- □ Ensure that you are using a GRiD Model 6201 cable, or equivalent, such as an INMAC null modem serial cable. See Serial Cable Requirements in this chapter for cable details.
- □ If more than one serial port exists on the IBM PC system, ensure that the card switches are set correctly and the cable is plugged into the correct serial port.
- □ Ensure that the serial port specified in the PCSLAVE command is the port that the cable is plugged into. If you are in doubt about how the ports are configured, try specifying the other port; e.g. if you entered PCLSAVE 1, try entering PCSLAVE 2.
- □ At the PCMASTER station, ensure that the CONFIG.SYS file on the boot device contains the statement DEVICE = PCMASTER.SYS [1 | 2]. Also, ensure that the serial port configuration hasn't been changed from the system default. (This can be changed by the MODE command, printer drivers, or communications programs.) You can reset the serial port by rebooting, or by the following commands: MODE SERIAL = ON and MODE COMn:9600,n,8,1.
- □ Some programs bypass the file system and access disks directly. Trying to access a slave device from the master station does not work when using one of these programs. For example, FORMAT, FDISK, DISKCOPY, and DISKCOMP will not recognize devices at the slave station.

Assignment of Device Letters

Device letters are assigned in alphabetical order, starting with the storage devices attached to the master station, and continue in sequence to those attached to the slave station. You can use the DEVICES command to check the device letters assigned to the slave devices. The following rules apply to the assignment of device letters.

□ Letters assigned to diskette drives at the master station alphabetically precede those assigned to hard disks and other devices.

□ The letters assigned to devices at the slave station follow in alphabetical order after the last letter assigned to the master station.

□ The letters assigned to GRiDCase devices follow the rules and precedences described in the Owner's Guide.

□ There are no logical device letters assigned to devices at the slave station.

Configuration Examples—GRiDCase Master Station

The following are examples of device letter assignments when you designate the GRiDCase as the master station.

Example 1 The master station (a GRiDCase) has only the internal diskette drive available; the slave station (IBM) has two diskette drives.

Master Station	Internal drive	A,B
Slave Station	Diskette drive 1	C
Slave Station	Diskette drive 2	D

Note: Two letters refer to the internal diskette drive, where the MS-DOS diskette usually resides. This lets you carry out commands that require a source and a target diskette in the internal drive. When you use the second drive designation letter, MS-DOS recognizes it as a "single-disk" operation.

Example 2 The master station (a GRiDCase) has two diskette drives and the slave station (an IBM) has one diskette drive.

Master Station	Internal drive	A,C
Master Station	External drive	B,D
Slave Station	Drive 1	E

Example 3 The master station (a GRiDCase) has one diskette drive and a hard disk; the slave station (an IBM) has one diskette drive and a hard disk.

Master Station	Internal drive	A,B
Master Station	Hard Disk	C
Slave Station	Drive 1	D
Slave Station	Hard Disk	E

Configuration Examples—IBM PC or PC/XT Master Station

The following are examples of device letter assignments when you designate the IBM PC or PC/XT as the master station.

Example 1 The master station (an IBM) has two diskette drives and the slave station (a GRiDCase) has one diskette drive.

Master Station	Drive 1	A
Master Station	Drive 2	B
Slave Station	Internal drive	C

Example 2 The master station (an IBM) has two diskette drives and a hard disk; the slave station (a GRiDCase) has one diskette drive and a hard disk.

Master Station	Drive 1	A
Master Station	Drive 2	B
Master Station	Hard Disk	C
Slave Station	Internal drive	D
Slave Station	Hard Disk	E

Chapter 12: The Linker Program

This chapter describes the Linker Program (LINK), which performs the following functions:

□ Combines 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.

□ Optionally produces a listing that shows both the resolution of external references and error messages.

Overview

When you write a program, you write it in source code. This source code is passed through a compiler which produces object modules. The object modules must be passed through the link process to produce machine language that the computer can understand directly. This machine language is in the form required for running programs.

You may wish to link (combine) several programs and run them together. Your programs may refer to symbols defined in other object 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 optionally produces a List file that shows external references resolved, and it also displays any error messages.

LINK uses available memory as much as possible. When available memory is exhausted, LINK creates a temporary disk file named VM.TMP.

How To Start LINK

LINK requires two types of input: a command to start LINK and responses to command prompts. In addition, several switches control LINK features. Usually, you will type all the commands to LINK on the terminal keyboard. As an option, answers to the command prompts and any switches may be contained in a response file.

LINK may be started in any of three ways. The first method is to type the commands in response to individual prompts supplied by LINK. In the second method, you type all commands on the line used to start LINK. In the third method, you must create a response file that contains all the necessary commands and tell LINK where that file is when you start LINK.

Summary of Methods to Start LINK

Method 1: LINK
Method 2: LINK filenames [/switches]
Method 3: LINK @filespec

Method 1: Prompts

To start LINK with Method 1, type: LINK (on a line by itself). LINK will be loaded into memory. LINK will then display four text prompts that appear one at a time. You answer the prompts to command LINK to perform specific tasks.

At the end of each line, you may type one or more switches, preceded by the switch character, a slash. Always end a line by pressing Return.

The command prompts are explained in the following table.

Prompt	**Responses**
Object Modules [.OBJ]:	List object files to be linked. The default filename extension is .OBJ. The filenames must be separated by blank spaces or plus signs (+). If a plus sign is the last character typed, the prompt will reappear. There is no default; a response is required.
Run File [Object-file.EXE]:	Give the filename for the executable object code produced by LINK. The default is first object filename with an extension of .EXE.
List File [NUL-file.MAP]:	Give the filename for the list file. The default filename extension is .MAP. The default filename is the NUL filename, i.e., the default produces no list file.
Libraries [.LIB]:	List filenames to be searched for routines or variables referenced but not defined in your program; separate the filenames by blank spaces or plus signs (+). If a plus sign is the last character typed, the prompt will reappear.

After you press Return for the last prompt, LINK begins linking automatically without further commands. When the link is finished, LINK exits to the operating system.

When entering filenames, you must give a pathname for any file that is not on the current drive and directory. If the linker cannot find an object file on the current disk, it displays a message and waits so that you can change disks if necessary.

You can use LINK options by typing them after the filename at any prompt.

At any prompt, you can type the rest of the filenames in the command line format described in the next section. For example, you can specify the default responses for all remaining prompts by typing a semicolon (;) after any prompt. When the linker encounters a semicolon, it immediately supplies the default responses and processes the remaining files without displaying any more prompts.

Example The following example shows how to start LINK by giving responses to individual prompts.

```
Object Modules [.OBJ]: MODA+MODB+
Object Modules [.OBJ]: MODC+STARTUP/PAUSE
Run File [moda.EXE]:
List file [NUL.MAP]: ABC
Libraries [.LIB]: B:\LIB\MATH
```

This example links the object modules MODA.OBJ, MODB.OBJ, MODC.OBJ, and STARTUP.OBJ. It searches the library file MATH.LIB on drive B: in the LIB directory for routines and data used in the program. It then creates an executable file named MODA.EXE, and a map file named ABC.MAP. The /PAUSE option in the Object Modules prompt line causes LINK to pause while you change disks before creating the executable file.

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 syntax:

LINK object-list [,runfile] [,listfile] [,lib-list] [/switches . . .]

where:

object-list is a list of object modules, separated by plus signs (+).

runfile is the name of the file to receive the executable output.

listfile is the name of the file to receive the listing.

lib-list is a list of library modules to be searched, separated by plus signs (+).

/switches refers to optional switches, which may be placed following any of the response entries (just before any of the commas or after the *lib-list*, as shown).

The commas (,) separating filenames for the different entry fields are required even if you don't supply a filename. To select the default for a field (the same as the base name of the first object file), simply type the comma that would follow the field without actually supplying a filename. You can use a semicolon (;) anywhere after the object file to terminate the command line. If you type the semicolon immediately after the object file, LINK supplies the default name for the runfile, produces no listfile, and searches no libraries.

If you do not supply all filenames in the command line and do not end the command line with a semicolon, the linker prompts for additional files using the prompts described in the previous section.

If you do not specify a drive or directory for a file, LINK assumes the file is on the current drive and directory.

NOTE: When linking modules that were produced by a high-level-language compiler that supports overlays (this does not include MASM), you must specify overlay modules by putting them in parentheses. Refer to your language compiler manual for details on specifying overlays.

Examples The following examples shows how to start LINK by using the command line.

```
LINK FILE.OBJ,FILE.EXE,FILE.MAP,ROUTINE.LIB
```

The first example is equivalent to the following example:

```
LINK FILE,,,ROUTINE
```

These two examples use the object file FILE.OBJ to create the executable file FILE.EXE. LINK searches the library ROUTINE.LIB for routines and variables used within the program. It also creates a list file called FILE.MAP.

```
LINK MODA MODB MODC STARTUP/PAUSE,,ABC,B:\LIB\MATH
```

The last example links the object modules MODA.OBJ, MODB.OBJ, MODC.OBJ, and STARTUP.OBJ. The linker searches through the library file MATH.LIB in the \LIB directory on drive B for routines and data used in the program. It then creates an executable file named MODA.EXE, and a map file named ABC.MAP. The /PAUSE option in the command line causes the linker to pause while you change disks before creating the executable file.

Method 3: Response File

To start LINK with Method 3, type:

```
LINK @filespec
```

where *filespec* is the name of a response file. A response file contains answers to the LINK prompts (shown in Method 1) and may also contain any of the switches. 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.

You can also specify a response file at any prompt, or at any position in a command line. The input from the response file is treated exactly as though it had been entered at prompts or in a command line.

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. The response to each prompt must be on a separate line, followed by a Return. If desired, a long response to the Object Modules: or Libraries: prompt may be typed on several lines by using a plus sign (+) to continue the same response onto the next line. If you do not supply a filename for a response, you must leave an empty line.

You use switches and command characters in the response file the same way you use them in responses typed on the keyboard.

You can place a semicolon on any line in the response file. When LINK encounters the semicolon, it automatically supplies default filenames for all files you have not yet named in the response file. The remainder of the response file is ignored.

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 filenames, the semicolon character, or carriage returns), LINK displays the prompt that does not have a response, and waits for you to type a legal response. When a legal response has been typed, LINK continues the link session.

Examples The following examples show how to start LINK by using a response file.

```
MODA MODB MODC STARTUP/PAUSE
ABC
B:\LIB\MATH
```

The response file above tells the linker to link the object modules MODA.OBJ, MODB.OBJ, MODC.OBJ, and STARTUP.OBJ. The linker searches through the library file MATH.LIB in the \LIB directory on drive B for routines and data used in the program. It then creates an executable file named MODA.EXE, and a map file named ABC.MAP. The /PAUSE option in the command line causes the linker to pause while you change disks before creating the executable file.

The following procedure combines all three methods of supplying filenames. Assume you have a response file called LIBRARY that contains one line:

```
LIB1+LIB2+LIB3+LIB4
```

Now stark LINK with a partial command line:

```
LINK OBJECT1 OBJECT2
```

LINK will then prompt you for the remaining files:

```
Run File [OBJECT1.EXE]: EXEC
List File [NUL.MAP]:
Libraries [.LIB]: @LIBRARY
```

You enter EXEC so that the linker will name the executable file EXEC.EXE. You then press Return to indicate that no map file is desired, and then enter @LIBRARY so that the linker will read in the response file containing the four library filenames.

Giving Search Paths for Libraries

You can direct LINK to search directories and disk drives for the libraries you have named in a command by specifying one or more search paths with the library names, or by assigning the search paths to the environment variable LIB before you invoke LINK. Setting environment variables is explained under the SET command in Chapter 7.

A search path is the path specification of a directory or drive name. You type search paths along with library names on the LINK command line or in response to the Libraries prompt. You can specify up to 16 search paths. You can also assign the search paths to the LIB environment variable, using the MS-DOS SET command. In the latter case, you must separate the search paths by semicolons (;).

If you include a drive or directory name in the filename for a library in the LINK command line, the linker searches there only. But if you don't give a drive or directory name, LINK searches for library files in the following order:

1. First, the linker searches the current drive and directory.
2. If the library is not found and one or more search paths have been given in the command line, the linker searches the specified search paths in the order in which you gave them.
3. If the library is still not found and you have set a search path by using the LIB environment variable, the linker searches there.
4. If the library is still not found, LINK prints an error message.

Examples

The following examples show how to specify search paths for libraries.

```
LINK FILE,,FILE,A:\ALTLIB\MATH.LIB+COMMON+B:+D:\LIB\
```

In the first example, the linker searches only the \ALTLIB directory on drive A to find the library, MATH.LIB, but to find COMMON.LIB it will search the current directory on the current drive, the current directory on drive B, and finally, the directory \LIB on drive D.

```
SET LIB=C:\LIB;D:\SYSTEM\LIB
LINK FILE,,FILE.MAP,MATH+COMMON
```

In the second example, LINK searches the current directory, the directory \LIB on drive C, and the directory \SYSTEM\LIB on drive D to find the libraries, MATH.LIB and COMMON.LIB.

The Map File

The map file lists the names, load addresses, and lengths of all segments in a program. It also lists the names and load addresses of any groups in the program, the program start address, and messages about any errors it may have encountered. If the /MAP option is used in the LINK command line, the map file lists the names and load addresses of all public symbols.

Segment information has the general form shown in this example:

```
Start       Stop        Length      Name        Class
00000H      0172CH      0172DH      TEXT        CODE
01730H      01E19H      006EAH      DATA        DATA
```

The Start and Stop columns show the 20-bit addresses (in hexadecimal) of the first and last byte in each segment. These addresses are relative to the beginning of the load module, which is assumed to be address 0000H. The operating system chooses its own starting address when the program is actually loaded. The Length column gives the length of the segment in bytes; the Name column, the name of the segment; and the Class column, the segment's class name.

Group information has the general form:

```
Origin        Group
0000:0        IGROUP
0173:0        DGROUP
```

In this example, IGROUP is the name of the code (instruction) group and DGROUP is the name of the data group.

At the end of the listing file, the linker gives you the address of the program entry point.

If you specify the /MAP option in the LINK command line, the linker adds a public-symbol list to the map file. The symbols are presented twice: once in alphabetical order, then in the order of their load addresses. The list has the general form shown in the following example:

```
Address       Publics by Name
0000:1567     BRK
0000:1696     CHMOD
0000:01DB     CHKSTK
0000:131C     CLEARERR
0173:0035     FAC

Address       Publics by Value
0000:01DB     CHKSTK
0000:131C     CLEARERR
0000:1567     BRK
0000:1696     CHMOD
0000:0035     FAC
```

The addresses of the public symbols are in segment:offset format. They show the location of the symbol relative to the beginning of the load module, which is assumed to be at address 0000:0000.

When the /HIGH and /DSALLOCATE options are used (see the section LINK Options) and the program's code and data combined do not exceed 64K, the map file may show symbols that have unusually large segment addresses. These addresses indicate a symbol whose location is below the actual start of the program code and data. For example, the symbol entry:

```
FFF0:0A20 TEMPLATE
```

shows that TEMPLATE is located below the start of the program. Note that the 20-bit address of TEMPLATE IS 00A20h.

The Temporary Disk File—VM.TMP

LINK normally uses available memory for the link session. If it runs out of available memory, it creates a temporary disk file named VM.TMP in the current working directory. When the linker creates this file, it displays the following message:

```
VM.TMP has been created. Do not change diskette in drive x
```

Note that *x* will be the proper drive name. After this message appears, you must not remove the disk from the drive specified by *x* until the link session ends. The /PAUSE option cannot be used if a temporary file is created. After LINK has created the executable file, it deletes the temporary file automatically.

NOTE: Do not use the filename VM.TMP for your own files. When the linker creates the temporary file, it destroys any previous file that has the same name.

LINK Options

The linker options, or switches, specify and control the tasks that LINK performs. All options begin with the linker-option character, which is the forward slash (/). You can use an option anywhere on a LINK command line.

LINK has the following options:

Option Name	Action
/HELP	Shows options list
/PAUSE	Pauses during linking
/EXEPACK	Packs executable file
/MAP	Creates public symbol map
/LINENUMBERS	Copies line numbers to map file
/NOIGNORECASE	Preserves case sensitivity in names
/NODEFAULTLIBRARYSEARCH	Overrides default libraries
/STACK	Sets maximum allocation space

/HIGH	Sets high load address
/DSALLOCATE	Allocates data group
/NOGROUPASSOCIATION	Sets group association override
/OVERLAYINTERRUPT	Sets overlay interrupt
/SEGMENTS	Sets maximum number of segments
/DOSSEG	Specifies MS-DOS segment ordering

You can abbreviate option names as long as your abbreviations contain enough letters to distinguish the specified option from other options. Minimum abbreviations are listed for each option.

Many of the LINK options set values in the MS-DOS program header. You will understand these options better if you understand how the header is organized. The program header is described in the *MS-DOS Programmer's Reference* and in some reference books on MS-DOS.

/HELP

The /HELP option causes LINK to write a list of the available options to the screen. This may be convenient if you need a reminder of the available options. You should not give a filename when using the /HELP option.

Minimum abbreviation: /HE

/PAUSE

The /PAUSE option causes LINK to pause before writing the executable file to disk so that you can change disks before the linker writes the executable (.EXE) file to disk.

If you specify the /PAUSE switch, the linker displays the following message before creating the run file:

```
About to generate .EXE file Change diskette in drive x and press <ENTER>
```

Note that *x* is the proper drive name. This message appears after the linker has read data from the object files and library files, and after it has written data to the map file, if you specified one. LINK resumes processing when you press Return. After LINK writes the executable file to disk, the following message appears:

```
Please replace original diskette in drive x and press <ENTER>
```

Minimum abbreviation: /P

NOTE: Do not remove the disk used for the VM.TMP file, if such a file has been created. If the temporary disk message appears when you have specified the /PAUSE option, you should press Ctrl-C to terminate the LINK session. Rearrange your files so that the temporary file and the executable file can be written to the same disk, then try again.

/EXEPACK

The /EXEPACK option directs LINK to remove sequences of repeated bytes (typically nulls) and optimize the load-time relocation table before creating the executable file. Executable files linked with the option may be smaller, and thus load faster than files linked without the option. However, the Microsoft Symbolic Debug Utility (SYMDEB) cannot be used with packed files.

The /EXEPACK option does not always save a significant amount of disk space (and may sometimes actually increase file size). Programs that have a large number of load-time relocations (about 500 or more) and long streams of repeated characters will usually be shorter if packed. If you're not sure if your program meets these conditions, try linking it both ways and compare the results.

Minimum abbreviation: /E

/MAP

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 that LINK creates. For a complete description of the listing-file format, see the section The Map File. The /MAP option is required if you want to use SYMDEB for symbolic debugging.

Minimum abbreviation: /M

NOTE: If you do not specify a map file in a LINK command, you can use the /MAP option to force the linker to create a map file. LINK gives the forced map file the same filename as the first object file specified in the command. It also adds the default extension .MAP.

/LINENUMBERS

The /LINENUMBERS option directs the linker to copy the starting address of each program source line to a map file. The starting address is actually the address of the first instruction that corresponds to the source line. You can use the MAPSYM program to copy line number data to a symbol file, which can then by used by SYMDEB.

The linker copies the line number data only if you give a map file name in the LINK command line, and only if the given object file has line number information.

If an object file has no line number information, the linker ignores the /LINENUMBERS option.

Minimum abbreviation: /LI

NOTE: If you do not specify a map file in a LINK command, you can still use the /LINENUMBERS option to force the linker to create a map file. Just place the option at or before the List File prompt. LINK gives the forced map file the same filename as the first object file that you specified in the command and gives it the default extension, .MAP.

/NOIGNORECASE

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. When you use the /NOIGNORECASE option, the linker treats TWO, two, and Two as different symbols.

Typically, you use the /NOIGNORECASE option with object files created by high-level-language compilers. Some compilers treat uppercase and lowercase letters as distinct letters and assume the linker does the same.

Minimum abbreviation: /NOI

/NODEFAULTLIBRARYSEARCH

The /NODEFAULTLIBRARYSEARCH option directs the linker 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 is linked with the program. Using this option overrides these default libraries and lets you explicitly name the libraries you want by including them on the LINK command line.

Minimum abbreviation: /NOD

/STACK:size

The /STACK option sets the program stack to the number of bytes given by *size*. The linker 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, the linker uses the given size in place of any value it may have calculated.

The size can be any positive integer value in the range 1 to 65535. The value can be a decimal, octal, or hexadecimal number. Octal numbers must begin with a zero. Hexadecimal numbers must begin with a leading zero followed by a lowercase x. For example, Ox1B.

By using the EXEMOD utility, you can also change the stack size after linking.

Minimum abbreviation: /ST

/CPARMAXALLOC:number

The /CPARMAXALLOC option sets the maximum number of 16-byte paragraphs needed by a program when it is loaded into memory. The operating system uses this number when allocating space for a program prior to loading it.

LINK normally sets the maximum number of paragraphs to 65535. Since this represents all addressable memory, the operating system always denies the request and allocates the largest contiguous block of memory it can find. If you use the /CPARMAXALLOC option, the operating system allocates no more space than given by this option. This means any additional space in memory is free for other programs.

The *number* can be any integer value in the range 1 to 65535. It must be a decimal, octal, or hexadecimal number. Octal numbers must begin with a zero. Hexadecimal values must begin with a leading zero followed by a lowercase x. For example, 0x2B.

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 needed. The minimum number of paragraphs needed by a program is never less than the number of paragraphs of code and data in the program.

Minimum abbreviation: /C

/HIGH

The /HIGH option sets a program's starting address to the highest possible address in free memory. If you don't use the /HIGH option, LINK sets the program's starting address as low as possible in memory.

Minimum abbreviation: /H

/DSALLOCATE

The /DSALLOCATE option directs the linker 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 you use /DSALLOCATE, 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.

Typically, you use the /DSALLOCATE option with the /HIGH option to take advantage of unused memory before the start of the program. The linker assumes that all free bytes in DGROUP occupy the memory preceding the program. To use the group, you must set a segment register to the start address of DGROUP.

Minimum abbreviation: /D

/NOGROUPASSOCIATION

The /NOGROUPASSOCIATION option directs LINK to ignore group associations when assigning addresses to data and code items.

Minimum abbreviation: /NOG

NOTE: This option exists strictly for compatibility with older versions of FORTRAN and Pascal (Microsoft version 3.13 or earlier, or any IBM version prior to 2.0). You should never use the /NOGROUPASSOCIATION option except to link with object files produced by those compilers, or with the run-time libraries that accompany the old compilers.

/OVERLAYINTERRUPT: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 0 to 255. It must be a decimal, octal, or hexadecimal number. Octal numbers must have a leading zero. Hexadecimal numbers must start with a leading zero followed by a lowercase x. For example, 0x3B.

You can only use this option if you are linking with a run-time module from a language compiler that supports overlays. Check your compiler documentation, since this option is not appropriate for use with some compilers.

Minimum abbreviation: /O

NOTE: You should not use interrupt numbers that conflict with the standard MS-DOS interrupts. See the *MS-DOS Programmer's Reference* for details.

/SEGMENTS:number

The /SEGMENTS option directs the linker to process no more than *number* segments per program. If it encounters more than the given limit, the linker displays an error message, and stops linking. You use this option to override the default limit of 128 segments.

If you do not use /SEGMENTS, the linker allocates enough memory space to process up to 128 segments. If your program has more than 128 segments, you will need to set the segment limit higher to increase the number of segments that LINK can process. If you get the following LINK error message:

```
Segment limit set too high
```

you should set the segment limit lower.

The number can be any integer value in the range 1 to 1024. It must be a decimal, octal, or hexadecimal number. Octal numbers must have a leading zero. Hexadecimal numbers must start with a leading zero followed by a lowercase x. For example, 0x4B.

Minimum abbreviation: /SE

/DOSSEG

The /DOSSEG option causes LINK to arrange all segments in the executable file according to the MS-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 other 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.

If you do not use the /DOSSEG option, see the section Groups under How Link Works for an explanation of the normal segment order.

Minimum abbreviation: /DO

How Link Works

LINK creates an executable file by concatenating a program's code and data segments according to the instructions in the original source files. These concatenated segments form an "executable image," that is copied directly into memory when you invoke the program for execution. Thus, the order and manner in which the linker copies segments to the executable file defines the order and manner in which it loads the segments into memory.

You can tell the linker how to link a program's segments by using a SEGMENT directive to supply segment attributes or by using the GROUP directive to form segment groups. These directives define group associations, classes, and align and combine types that define the order and relative starting addresses of all segments in a program. This information works in addition to any information you supply through command-line options.

The following sections explain the process that LINK uses to concatenate segments and resolve references to items in memory.

Alignment of Segments

The linker uses a segment's align type to set the starting address for the segment. The align types are byte, word, para, and page. These types correspond to starting addresses at byte, word, paragraph, and page boundaries, representing addresses that are multiples of 1, 2, 16, and 256, respectively. The default align type is para.

When the linker encounters a segment, it checks the align type before copying the segment to the executable file. If the align type is word, para, or page, the linker checks the executable image to see if the last byte copied ends at an appropriate boundary. If not, LINK pads the image with extra null bytes.

Frame Number

The linker computes a starting address for each segment in a program. The starting address is based on a segment's align type and the size of the segments already copied to the executable file. The address consists of an offset and a canonical frame number, which specifies the address of the first paragraph in memory that contains one or more bytes of the segment. A frame number is always a multiple of 16 (a paragraph address). But the offset is the number of bytes from the start of the paragraph to the first byte in the segment. For byte and word align types, the offset may be nonzero. The offset is always zero for para and page align types.

The frame number of a segment can be obtained from a LINK file. The frame number is the first five hexadecimal digits of the start address specified for the segment.

Order of Segments

LINK copies segments to the executable file in the same order that it encounters them in the object files. The linker maintains this order throughout the program unless it encounters two or more segments with the same class name. Segments with identical class names belong to the same class type, and are copied to the executable file as contiguous blocks.

For a more detailed discussion of segment loading order and methods of controlling loading order by assigning class types, see the *Microsoft Macro Assembler Reference Manual*.

Combined Segments

LINK uses combine types to determine whether two or more segments sharing the same segment name should be combined into a single, large segment. The combine types are public, stack, common, memory, at, and private. Combine types are also described in the *Microsoft Macro Assembler Reference Manual*.

If a segment has combine type public, the linker automatically combines it with any other segments that have the same name and belong to the same class. When LINK combines segments, it ensures that the segments are contiguous and that all addresses in the segments can be accessed using an offset from the same frame address. The result is the same as if the segment were defined as a whole in the source file.

The linker preserves each segment's align type. This means that even though the segments belong to a single, large segment, the code and data in the segments retain their original align type. If the combined segments exceed 64K, LINK displays an error message.

If a segment has combine type stack, the linker carries out the same combine operation as for public segments. The only difference is that stack segments cause LINK to copy an initial stack-pointer value to the executable file. This stack-pointer value is the offset to the end of the first stack segment (or combined stack segment) that the linker encounters. If you use the stack type for stack segments, you do not need to give instructions that load the segment into the SS register.

If a segment has combine type common, the linker combines it automatically with any other segments that have the same name and belong to the same class. When LINK combines common segments, however, it places the start of each segment at the same address, creating a series of overlapping segments. The result is a single segment which is no larger than the largest of the combined segments.

The linker treats segments with combine type memory exactly like segments with combine type public. MASM provides combine type memory for compatibility with linkers that support a separate combine type for memory segments.

A segment has combine type private only if no explicit combine type is defined for it in the source file. LINK does not combine private segments.

Groups

Groups permit non-contiguous segments that do not belong to the same class to be addressable relative to the same frame address. When LINK encounters a group, it adjusts all memory references to items in the group so that they are relative to the same frame address.

Segments in a group do not have to be contiguous, do not have to belong to the same class, and do not have to have the same combine type. The only requirement is that all segments in the group fit within 64K.

Groups do not affect the order in which the segments are loaded. Unless you use class names and enter object files in the right order, there is no guarantee that the segments will be contiguous. In fact, the linker may place segments that do not belong to the group in the same 64K of memory. Although LINK does not explicitly check that all segments in a group fit within 64K of memory, the linker is likely to encounter a "fixup-overflow" error if this requirement is not met.

Groups, and how to define them, are discussed in the *Microsoft Macro Assembler Reference Manual*.

Fixups

Once the starting address of each segment in a program is known, and all segment combinations and groups have been established, the linker can fix up any unresolved references to labels and variables. To fix up unresolved references, the linker computes an appropriate offset and segment address and replaces the temporary values, generated by the assembler, with the new values.

LINK carries out fixups for four different references:

□ Short

□ Near self-relative

□ Near segment-relative

□ Long

The size of the value to be computed depends on the type of reference. If LINK discovers an error in the anticipated size of a reference, it displays a fixup-overflow message. This can happen, for example, if a program attempts to use a 16-bit offset to reach an instruction in a segment that has a different frame address. It can also occur if all segments in a group do not fit within a single 64K block of memory.

A short reference occurs in JMP instructions that attempt to pass control to labeled instructions that are in the same segment or group. The target instruction must be no more than 128 bytes from the point of reference. The linker computes a signed, 8-bit number for this reference and displays an error message if the target instruction belongs to a different segment or group (has a different frame address), or if the target is more than 128 bytes distant (in either direction).

A near self-relative reference occurs in instructions which access data relative to the same segment or group. The linker computes a 16-bit offset for this reference and displays an error message if the data are not in the same segment or group.

A near segment-relative reference occurs in instructions which attempt to access data that is in a specified segment or group, or that is relative to a specified segment register. LINK computes a 16-bit offset for this reference and displays an error message if the offset of the target within the specified frame is greater than 64K or less than 0, or if the beginning of the canonical frame of the target is not addressable. A long reference occurs in CALL instructions that attempt to access an instruction in another segment or group. LINK computes a 16-bit frame address and 16-bit offset for this reference. The linker displays an error message if the computed offset is greater than 64K or less than 0, or if the beginning of the canonical frame of the target is not addressable.

Chapter 13: The DEBUG Utility

This chapter describes the DEBUG Utility which is a debugging program that provides a controlled testing environment for binary and executable object files. Note that EDLIN can be used to alter source files; DEBUG is EDLIN's counterpart for binary files. DEBUG eliminates the need to reassemble a program to see if a problem has been fixed by a minor change. It allows you to alter the contents of a file or the contents of a CPU register, and then to immediately reexecute a program to check on the validity of the changes.

All DEBUG commands may be aborted at any time by pressing Ctrl-C. Ctrl-S suspends the display, so that you can read it before the output scrolls away. Entering any key other than Ctrl-C or Ctrl-S restarts the display. All of these commands are consistent with the control character functions available at the MS-DOS command level.

How to Start DEBUG

DEBUG may be started two ways. By the first method, you type all commands in response to the DEBUG prompt (a hyphen). By the second method, you type all commands on the line used to start DEBUG.

Summary of Methods to Start DEBUG:

Method 1: DEBUG

Method 2: DEBUG [filespec [arglist]]

Method 1: DEBUG

To start DEBUG using method 1, type:

```
DEBUG
```

DEBUG responds with the hyphen (-) prompt, signaling that it is ready to accept your commands. Since no filename has been specified, current memory, disk sectors, or disk files can be worked on by using other commands.

CAUTION: When DEBUG (Version 2.0) is started, it sets up a program header at offset 0 in the program work area. On previous versions of DEBUG, you could overwrite this header. You can still overwrite the default header if 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 will terminate.

CAUTION: Do not restart a program after the "Program terminated normally" message is displayed. You must reload the program with the N and L commands for it to run properly.

Method 2: Command Line

To start DEBUG using a command line, type:

DEBUG [filespec [arglist]

For example, if a *filespec* is specified, then the following is a typical command to start DEBUG:

```
DEBUG FILE.EXE
```

DEBUG then 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.

An *arglist* may be specified if *filespec* is present. The *arglist* is a list of filename parameters and switches that are to be passed to the program *filespec*. Thus, when *filespec* is loaded into memory, it is loaded as if it had been started with the command:

filespec arglist

Here, *filespec* is the file to be debugged, and the *arglist* is the rest of the command line that is used when *filespec* is invoked and loaded into memory.

Commands and Parameters

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 the EDLIN section apply inside DEBUG.

All DEBUG commands accept parameters, except the Quit command. Parameters may be separated by 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
```

If a syntax error occurs in a DEBUG command, DEBUG reprints the command line and indicates the error with an up-arrow (↑) and the word "error."

For example:

```
dcs:100 cs:110
     ↑ error
```

Any combination of uppercase and lowercase letters may be used in commands and parameters.

The DEBUG commands are summarized in Table 13-1 and are described in detail, with examples, following the description of command parameters. The parameters available for use with DEBUG commands are described in Table 13-2.

Table 13-1. *DEBUG Commands*

Command	**Function**
A [address]	Assemble
C range address	Compare
D [range]	Dump
E address [list]	Enter
F range list	Fill
G [= address [address . . .]]	Go
H value value	Hex
I value	Input
L [address [drive:record record]]	Load
M range address	Move
N filename [filename]	Name
O value byte	Output
Q	Quit
R [register-name]	Register
S range list	Search
T [= address] [value]	Trace
U [range]	Unassemble
W [address [drive:record record]]	Write

Table 13-2. *DEBUG Parameters*

Parameter	Definition
drive:	A one-digit hexadecimal value to indicate which drive a file will be loaded from or written to. The valid values are 0-3. These values designate the drives as follows: 0 = A:, 1 = B:, 2 = C:, 3 = D:.
byte	A two-digit hexadecimal value to be placed in or read from an address or register.
record	A 1- to 3-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 may 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.

range	Two *address*es: e.g., *address address*; or one *address*, an L, and a *value*: e.g., *address* L *value* where *value* is the number of lines the command should operate on, and L80 is assumed. The last form cannot be used if another hex value follows the *range*, since the hex 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:100`
	`  ↑ error`
	The limit for *range* is 10000 hex. To specify a *value* of 10000 hex within four digits, type 0000 (or 0).

Parameter	Definition
list	A series of *byte* values or of *string*s. *list* must be the last parameter on the command line. Example: fcs:100 42 45 52 54 41
string	Any number of characters enclosed in quote marks. Quote marks may be either single (') or double("). If the delimiter quote marks must appear within a *string*, the quote marks must be doubled. For example, the following strings are legal: 'This is a "string" is okay.' 'This is a ''string'' is okay.' However, this string is illegal: 'This is a 'string' is not.' Similarly, these strings are legal: "This is a 'string' is okay." "This is a ""string"" is okay." However, this string is illegal: "This is a "string" is not." Note that the double quote 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.

Assemble

The Assemble command assembles 8086/8087/8088 mnemonics directly into memory.

Assemble has the following format:

A [address]

If a syntax error is found, DEBUG responds with

```
↑Error
```

and redisplays the current assembly address.

All numeric values are hexadecimal and must be entered as 1-4 characters. Prefix mnemonics must be specified in front of the opcode to which they refer. They may also be entered 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 will automatically assemble short, near or far jumps and calls, depending on byte displacement to the destination address. These may 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 may 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, the data type must be explicitly stated 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 will assemble byte values directly into memory. The DW opcode will assemble 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
```

Compare

The Compare command compares the portion of memory specified by *range* to a portion of the same size beginning at *address*.

Command has the following format:

C range address

If the two areas of memory are identical, there is no display and DEBUG returns with the MS-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

```
C100L100 300
```

Each command compares the block of memory from 100 to 1FFH with the block of memory from 300 to 3FFH.

Dump

The Dump command displays the contents of the specified region of memory.

Dump has the following format:

```
D [range]
```

If a range of addresses is specified, the contents of the range are displayed. If the D command is typed without parameters, 128 bytes are displayed at the first address (DS:100) after the address 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. Each displayed line begins on a 16-byte boundary.

Examples

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 the following command:

```
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 the command:

```
DCS:100 L 20
```

the display is formatted as described above, but 20H bytes are displayed.

If then you type the command:

```
DCS:100 115
```

the display is formatted as described above, but all the bytes in the range of lines from 100H to 115H in the CS segment are displayed.

Enter

The Enter command enters byte values into memory at the specified *address*.

Enter has the following format:

E address [list]

If the optional *list* of values is typed, the replacement of byte values occurs automatically. (If an error occurs, no byte values are changed.)

If the *address* is typed without the optional *list*, DEBUG displays the address and its contents, then repeats the address on the next line and waits for your input. At this point, the Enter command waits for you to perform one of the following actions:

1. Replace a byte value with a value you type. Simply 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 Spacebar to advance to the next byte. To change the value, simply type the new value as described in (1.) above. 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, typing the hyphen returns the current position to the previous byte. When the hyphen is typed, a new line is started with the address and its byte value displayed.
4. Press Return to terminate the Enter command. Return may be pressed at any byte position.

Examples

Assume that the following command is typed:

```
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 Spacebar 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_
```

Pressing Return ends the Enter command and returns to the DEBUG command level.

Fill

The Fill command fills the addresses in the *range* with the values in the *list*. Fill has the following format:

F range list

If the *range* contains more bytes than the number of values in the *list,* the *list* will be used repeatedly until all bytes in the *range* are filled. If the *list* contains more values than the number of bytes in the *range,* the extra values in the *list* will be ignored. If any of the memory in the *range* is not valid (bad or nonexistent), an error will occur in all succeeding locations.

Example

Assume that the following command is typed:

```
F04BA:100 L 100 42 45 52 54 41
```

DEBUG fills memory locations 04BA:100 through 04BA:1FF with the bytes specified. The five values are repeated until all 100H bytes are filled.

Go

The Go command executes the program currently in memory.

Go has the following format:

```
G[ = address[ address . . . ]]
```

If only the Go command is typed, 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 *address*es.

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 ten breakpoints may be set. Breakpoints may be set only at addresses containing the first byte of an 8086 opcode. If more than ten 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 G 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. (Beware, if the user stack is not valid or is too small, the operating system may crash.) An interrupt code (0CCH) 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

Assume that the following command is typed:

```
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, then the program executes just as if you had typed the filename at the MS-DOS command level. The only difference is that program execution begins at the instruction after the breakpoint rather than at the usual start address.

Hex

The Hex command performs hexadecimal arithmetic on the two parameters specified.

Hex has the following format:

H value value

First, DEBUG 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

Assume that the following command is typed:

```
H19F 10A
```

DEBUG performs the calculations and then displays the result:

```
02A9  0095
```

Input

The Input command inputs and displays one byte from the port specified by *value*.

Input has the following format:

I value

A 16-bit port address is allowed.

Example

Assume that you type the following command:

```
I2F8
```

Assume also that the byte at the port is 42H. DEBUG inputs the byte and displays the value:

```
42
```

Load

The Load command loads a file into memory.

Load has the following format:

```
L [address [drive:record record]]
```

Set BX:CX 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 filename properly in the normal format of a file control block at CS:5C.

If the L command is typed 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 the L command is typed with an address parameter, loading begins at the memory *address* specified. If L is typed 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

Assume that the following commands are typed:

```
A>DEBUG
-NFILE.COM
```

Now, to load FILE.COM, type:

```
L
```

DEBUG loads the file and then displays the DEBUG prompt. Assume that you want to load only portions of a file or certain records from a disk. To do this, type:

```
L04BA:100 2 0F 6D
```

DEBUG then loads 109 (6D hex) records beginning with logical record number 15 into memory beginning at address 04BA:0100. When the records have been loaded, DEBUG simply returns the 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 will differ from its size in memory.

If the file named by the Name command or specified when DEBUG is started is a .HEX file, then typing the L command with no parameters causes DEBUG to load the file beginning at the address specified in the .HEX file. If the L command includes the option *address*, DEBUG adds the *address* specified in the L command to the address found in the .HEX file to determine the start address for loading the file.

Move

The Move command moves the block of memory specified by *range* to the location beginning at the *address* specified.

Move has the following format:

```
M range address
```

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 towards the lowest.

Note that if the addresses in the block being moved will not have new data written to them, the data there before the move will remain. The M 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

Assume that 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 D command, using the *address* typed for the M command, to review the results of the move.

Name

The Name command sets filenames.

Name has the following format:

```
N filename [filename . . . ]
```

The Name command performs two functions. First, Name is used to assign a filename for a later Load or Write command. Thus, if you start DEBUG without naming any file to be debugged, then the N*filename* command must be typed before a file can be loaded. Second, Name is used to assign filename 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
```

Name will perform the following steps:

1. Name assigns the filename FILE1.EXE to the filename to be used in any later Load or Write commands.
2. Name also assigns the filename FILE1.EXE to the first filename parameter used by any program that is later debugged.
3. Load loads FILE1.EXE into memory.
4. Go causes FILE1.EXE to be executed with FILE1.EXE as the single filename parameter (that is, FILE1.EXE is executed as if FILE1.EXE had been typed at the command level).

A more useful chain of commands might look like this:

```
-NFILE1.EXE
-L
-NFILE2.DAT FILE3.DAT
-G
```

Here, Name sets FILE1.EXE as the filename for the subsequent Load command. The Load command loads FILE1.EXE into memory, and then the Name command is used again, this time to specify the parameters to be used by FILE1.EXE. Finally, when the Go command is executed, FILE1.EXE is executed as if FILE1 FILE2.DAT FILE3.DAT had been typed at the MS-DOS command level. Note that if a Write command were executed at this point, then 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.

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 filename parameter given to the Name command is set up at CS:5C. If a second filename parameter is typed, then 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 MS-DOS command level.

Example

A typical use of the Name command is shown in the following example:

```
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 runtime environment for PROG.COM.

Output

The Output command sends the *byte* specified to the output port specified by *value*.

Output has the following format:

O value byte

A 16-bit port address is allowed.

Example

Type:

```
02F8 4F
```

DEBUG outputs the byte value 4F to output port 2F8.

Quit

The Quit command terminates the DEBUG utility.

Quit has the following format:

```
Q
```

The Q command takes no parameters and exits DEBUG without saving the file currently being operated on. You are returned to the MS-DOS command level.

Register

The Register command displays the contents of one or more CPU registers.

Register has the following format:

```
R [register-name]
```

If no *register-name* is typed, the R command dumps the register save area and displays the contents of all registers and flags.

If a register name is typed, the 16-bit value of that register is displayed in hexadecimal, and then a colon appears as a prompt. You then either type a *value* to change the register, or simply press Return if no change is wanted.

The only valid *register-name*s are:

AX
BX
CX
DX
SP
BP
SI
DI
DS
ES
SS
CS
IP
PC
F

(IP and PC both refer to the Instruction Pointer.)

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 alphabetic code. To alter any flag, type the opposite two-letter code. The flags are either set or cleared.

The flags are listed in Table 13-3 with their codes for SET and CLEAR.

Table 13-3. *Flag SET and CLEAR Codes*

Flag Name	Set	Clear
Overflow	OV	NV
Direction	DN Decrement	UP Increment
Interrupt	EI Enabled	DI Disabled
Sign	NG Negative	PL Plus
Zero	ZR	NZ
Auxiliary Carry	AC	NA
Parity	PE Even	PO Odd
Carry	CY	NC

Whenever you type the 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 may enter new flag values as alphabetic pairs. The new flag values can be entered in any order. You do not have to leave spaces between the flag entries. To exit the R command, press Return. Flags for which new values were not entered remain unchanged.

If more than one value is entered 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 startup, the segment registers are set to the bottom of free memory, the Instruction Pointer is set to 0100H, all flags are cleared, and the remaining registers are set to zero.

Examples

Type:

```
R
```

DEBUG displays all registers, flags, and the decoded instruction for the current location. If the location is CS:11A, then the display will look similar to this:

```
AX=0E00 BX=00FF CX=0007 DX=01FF SP=039D BP=0000
SI=005C DI=0000 DS=04BA ES=04BA SS=04BA CS=04BA
IP=011A NV UP DI NG NZ AC PE NC
04BA:011A CD21     INT  21
```

If you type:

```
RF
```

DEBUG will display the flags:

```
NV UP DI NG NZ AC PE NC - _
```

Now, type any valid flag designation, in any order, with or without spaces. For example:

```
NV UP DI NG NZ AC PE NC - PLEICY
```

DEBUG responds only with the DEBUG prompt. To see the changes, type either the R or RF command:

```
RF
NV UP EI PL NZ AC PE CY - _
```

Press Return to leave the flags this way, or specify different flag values.

Search

The Search command searches the *range* specified for the *list* of bytes specified.

Search has the following format:

S range list

The *list* may 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 this:

```
04BA:0104
04BA:010D
-type:
```

Trace

The Trace command executes one instruction and displays the contents of all registers and flags, and the decoded instruction.

Trace has the following format:

T [= address][value]

If the optional = *address* is typed, tracing occurs at the = *address* specified. The optional *value* causes DEBUG to execute and trace the number of steps specified by *value*.

The Trace command uses the hardware trace mode of the 8086 or 8088 microprocessor. Consequently, you may also trace instructions stored in ROM (Read Only Memory).

Examples

Type:

```
T
```

DEBUG returns a display of the registers, flags, and decoded instruction for that one instruction. Assume that the current position is 04BA:011A; DEBUG might return the display:

```
AX=0E00 BX=00FF CX=0007 DX=01FF SP=039D BP=0000
SI=005C DI=0000 DS=04BA ES=04BA SS=04BA CS=04BA
IP=011A NV UP DI NG NZ AC PE NC
04BA:011A CD21      INT  21
```

If you type:

```
T=011A 10
```

DEBUG executes sixteen (10 hex) instructions beginning at 011A in the current segment, and then displays all registers and flags for each instruction as it is executed. The display scrolls 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.

Unassemble

The Unassemble command disassembles bytes and displays the source statements that correspond to them, with addresses and byte values.

Unassemble has the following format:

U [range]

The display of disassembled code looks like a 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, then DEBUG disassembles all bytes in the range. If the *range* is given as an *address* only, then 20H bytes are disassembled.

Examples

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 shows:

```
04BA:0100      206472      AND      [SI+72],AH
04BA:0103      69          DB       69
04BA:0104      7665        JBE      016B
04BA:0106      207370      AND      [BP+DI+70],DH
```

If the bytes in some addresses are altered, the disassembler alters the instruction statements. The U command can be typed for the changed locations, the new instructions viewed, and the disassembled code used to edit the source file.

Write

The Write command writes the file being debugged to a disk file.

Write has the following format:

```
W [address[ drive:record record]]
```

If you type W with no parameters, BX:CX must already be set to the number of bytes to be written; the file is written beginning from CS:100. If the W command is typed with just an address, then the file is written beginning at that address. If a G or T command has been used, BX:CX must be reset before using the Write command without parameters. Note that 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 invocation command or with the N command (refer to the Name command earlier in this manual). Both the DEBUG invocation and the N command format a filename properly in the normal format of a file control block at CS:5C.

If the W command is typed with parameters, the write begins from the memory address specified; the file is written to the *drive:* specified (the drive designation is numeric here—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.

CAUTION: Writing to absolute sectors is EXTREMELY dangerous because the process bypasses the file handler.

Examples

Type:

```
W
```

DEBUG will write the file to disk and then display 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
-_
```

Error Messages

During the DEBUG session, you may receive any of the following error messages. Each error terminates the DEBUG command under which it occurred, but does not terminate DEBUG itself.

BF

Bad flag. You attempted to alter a flag, but the characters typed were not one of the acceptable pairs of flag values. See the Register command for the list of acceptable flag entries.

BP

Too many breakpoints. You specified more than ten breakpoints as parameters to the G command. Retype the Go command with ten or fewer breakpoints.

BR

Bad register. You typed the R command with an invalid register name. See the Register command for the list of valid register names.

DF

Double flag. You typed two values for one flag. You may specify a flag value only once per RF command.

Messages

This section provides a comprehensive alphabetical list of all message you may receive under MS-DOS.

Abort edit (Y/N)?

EDLIN. This message is displayed when you choose the Q (Quit) command in EDLIN. The Quit command exits the editing session without saving any editing changes. Specify Y (for Yes) or N (for No).

Abort, Retry, Ignore?

MS-DOS. An error has occurred on a diskette, hard disk, or some other device. First, try to correct the condition that caused the error. Then, enter a response in the following order:

R Retry. MS-DOS attempts to execute the command or program that caused the error.

A Abort. MS-DOS terminates the command or program that caused the error.

I Ignore. MS-DOS ignores the error and tries to continue processing as if no error occurred.

Access denied

REPLACE, XCOPY. You tried to replace or copy a write-protected, read-only, or locked file.

All files cancelled by operator

PRINT. This message is displayed when you specify the /T switch with the PRINT command.

All specified files are contiguous

CHKDSK. All files are allocated contiguously on the disk without fragmentation.

Allocation error in file, size adjusted

CHKDSK. An invalid sector number was found in the File Allocation Table (an internal table used by MS-DOS to manage files). The file was truncated at the end of the last valid cluster.

Are you sure (Y,N)?

MS-DOS. MS-DOS displays this message if you try to delete *.* (all files in the current directory). Specify Y (for Yes) or N (for No).

Attempted write-protect violation

FORMAT. The disk to be formatted is write-protected.

Bad call format reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Bad call format writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Bad command error reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Bad command error writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Bad command or file name

MS-DOS. The command cannot find the file you asked it to run. Correct this error as follows:

- □ Make sure that you typed in the command name correctly; if not reenter the command and press Return.
- □ Using the Directory command (DIR), make sure that the current directory contains a file with the same name as the command you typed in. If not, either
  - – Change to a directory that contains the file or
  - – For MS-DOS commands, copy in the file from your backup system diskette.

  Reenter the command and press Return.

Bad or missing *filename*

MS-DOS. The Device statement in the CONFIG.SYS file contains an incorrect device specification. Check the accuracy of the DEVICE command in the CONFIG.SYS file.

Bad or missing Command Interpreter

MS-DOS. Either the COMMAND.COM file doesn't exist on the boot device, or the existing file is invalid. Re-boot the system using your backup MS-DOS diskette, then copy the COMMAND.COM file onto the device you tried to boot from in the first place.

NOTE: You may also receive this message when, after the booting MS-DOS, you later delete COMMAND.COM from the boot device.

Bad Partition Table

FORMAT. There is no MS-DOS partition on the hard disk. You must first run FDISK to create an MS-DOS paritition before you can format it.

Bad unit error reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Bad unit error writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

BREAK is off (or on)

MS-DOS. This message tells you the current setting of BREAK.

Cannot CHDIR to *filename*—tree past this point not processed

CHKDSK. CHKDSK is traveling the tree structure of the directory and is unable to proceed to the specified directory. All subdirectories beneath this directory will not be verified.

Cannot CHDIR to root
Processing cannot continue

CHKDSK. CHKDSK is traveling the tree structure of the directory and is unable to return to the root directory. CHKDSK is not able to continue checking the remaining subdirectories to the root.

Cannot CHKDSK a Network drive

CHKDSK. You cannot check drives that are redirected over the network.

Cannot CHKDSK a SUBSTed or ASSIGNed drive

CHKDSK. You cannot check drives that have been substituted or assigned.

Cannot DISKCOMP to or from a network drive

DISKCOMP. You cannot compare disks on drives that have been redirected over the network.

Cannot DISKCOMP to or from an ASSIGNed or SUBSTed drive

DISKCOMP. You cannot compare to a drive that has been created using the ASSIGN or SUBST commands.

Cannot DISKCOPY to or from a network drive

DISKCOPY. You cannot copy disks to or from drives that have been redirected over the network.

Cannot do binary reads from a device

MS-DOS. This message appears during COPY command processing. The COPY cannot be done in binary mode when you are copying from a device. Remove the /B switch or specify an ASCII copy with the /A switch.

Cannot edit .BAK file—rename file

EDLIN. You attempted to edit a backup copy created by EDLIN. Either rename the file or copy the .BAK file and give it a different extension.

Cannot format an ASSIGNed or SUBSTed drive

FORMAT. The drive has been assigned or substituted and you attempted to format it. Restore the default drive assignments, then format the drive.

Cannot FORMAT a Network drive

FORMAT. You cannot format drives that are redirected over the network.

Cannot label a network drive

You cannot label a drive that is shared on a network server station.

Cannot open *filename*

PRINT. Either MS-DOS cannot find the specified file to print or the file does not exist. Check the command for a valid filename.

Cannot perform a cyclic copy

XCOPY. When you are using the /S switch, you cannot specify a target that is a subdirectory of the source.

Cannot recover . entry, processing continued

CHKDSK. The . entry (current directory) is defective.

Cannot recover .. entry

CHKDSK. The .. entry (parent directory) is defective.

Cannot Recover a Network drive

RECOVER. You cannot recover files on drives that are redirected over the network.

Cannot SUBST a network drive

SUBST. You cannot substitute drives that are redirected over the network.

Cannot SYS to a Network drive

SYS. You cannot transfer the system files to drives that are redirected over the network.

Cannot use PRINT—Use NET PRINT

PRINT. You must use the NET PRINT command to print files.

Cannot XCOPY to a reserved device

XCOPY. You cannot copy files to the device you specified.

Cannot XCOPY from a reserved device

XCOPY. You cannot copy files from the device you specified.

CHDIR .. failed, trying alternate method

CHKDSK. In traveling the tree structure, CHKDSK was not able to return to a parent directory. It will try to return to that directory by starting over at the root and traveling down.

Compare another diskette (Y/N)?

DISKCOMP. Diskcomp displays this message when it has completed is comparison of the disks. Press Y if you want to compare more disks.

Compare error(s) on side *sss*, track *ttt*

DISKCOMP. Data on the side and track indicated was not identical, but the data on the rest of the diskette compares.

Compare OK

DISKCOMP. The disks are identical.

Compare process ended

DISKCOMP. A fatal error occurred during the disk comparison.

Comparing *tt* tracks *ss* sectors per track, *y* side(s)

DISKCOMP. This message confirms the format of the disk that you are comparing.

COM port does not exist.

MODE. You specified an invalid COM port.

Content of destination lost before copy

MS-DOS. A file to be used as a source file to the COPY command has been overwritten prior to completion of the copy. Example: COPY A + B B which destroys B before it can be copied.

Convert lost chains to files (Y/N)?

CHKDSK. If you respond Y to this prompt, CHKDSK will recover the lost blocks it found when checking the disk. CHKDSK will create a directory entry and a file for you with the filename FILEnnnn.CHK. If you respond N, CHKDSK frees the lost blocks so they can be reallocated.

Copy another (Y/N)?

DISKCOPY. Respond Y if you wish to copy another disk. Respond N if you do not wish to copy another disk.

Copy complete

DISKCOPY. DISKCOPY has completed processing.

Copy not completed

DISKCOPY. DISKCOPY could not copy the entire disk.

Copying . . .

DISKCOPY. This message indicates that DISKCOPY is copying a disk.

Corrections will not be written to disk

CHKDSK. There are errors on the disk, but you did not specify the /F switch to correct them.

Data error reading drive *x:*

MS-DOS. Device error. (*x* is a letter indicating the storage device.) Take the action described under the message Abort, Retry, Ignore?

PCMASTER. You issued a command for a PC file transfer. Make sure the null modem cable connecting the serial ports of the two computers is firmly inserted and try again. Then, enter R (for *retry*).

Data error writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Device Loaders Not Loaded from MS-DOS Make sure CONFIG.SYS is on Boot Disk

You attempted to use a command or program that requires information in the CONFIG.SYS file provided by MS-DOS. To correct the error, copy MS-DOS from your backup system diskette to the boot device, and re-boot your system.

Delete current volume label (Y/N)?

LABEL. When you press the Return key in response to the prompt to enter the new volume label for this disk, LABEL displays this message if a current volume label exists. Press Y (for Yes) if you want to delete the volume label, otherwise, press N (for No).

DEVICE Support Not Present

DISKCOMP. The disk drive does not support MS-DOS 3.2 device control.

Directory is joined

CHKDSK. CHKDSK cannot process directories which are joined.

Directory is totally empty, no . or ..

CHKDSK. The specified directory does not contain references to current and parent directories. Delete the specified directory and recreate it.

Directory not empty

JOIN. You can only join onto a directory which is empty.

Disk error reading drive *x:*

MS-DOS. Device error. If *x:* is a floppy disk drive, make sure the diskette is inserted correctly and the latch is secured. Take the action described under the message Abort, Retry, Ignore?

Disk error reading FAT *x*

CHKDSK. A File Allocation Table (FAT, an internal table MS-DOS uses in managing files) has a defective sector. MS-DOS automatically uses another FAT. Copy your files to a new disk; reinitialize the old disk with the Format command before using it again.

Disk error writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Disk error writing FAT *x:*

CHKDSK. A File Allocation Table (FAT, an internal table MS-DOS uses in managing files) has a defective sector. MS-DOS automatically uses another FAT. Copy your files to a new disk; reinitialize the old disk with the Format command before using it again.

Diskettes compare OK

DISKCOMP. The disks you compared are identical.

Disk full. Edits lost

EDLIN. EDLIN was not able to save your file due to lack of disk space.

Disk full—write not completed

EDLIN. You gave the END command, but the disk did not contain enough free space for the file. EDLIN aborted the E command and returned you to the operating system. Part of the file may have been written to disk and saved. Delete the saved portion and restart the editing session. The file will not be available after this error.

Disks must be the same size

DISKCOPY. You cannot copy the contents of a disk with a different format using DISKCOPY. Use the COPY or XCOPY command to copy files onto the disk.

Disk unsuitable for system disk

FORMAT. FORMAT detected a bad track on the device where the MS-DOS files MSDOS.SYS and IO.SYS must reside. This device cannot be used as the boot device.

If formatting a diskette, try the command again using a new diskette; the old diskette can be used for storing data, but not as a boot diskette.

If formatting a hard disk, contact your GRiD representative.

Divide error

ATTRIB. The 8086 has set the divide overflow flag which is usually caused by division by zero.

Divide overflow

MS-DOS. The 8086 has set the divide overflow flag which is usually caused by division by zero.

Does *name* specify a file name
or directory name on the target
(F = file D = directory)?

XCOPY. XCOPY displays this prompt if the target directory does not exist.

Do not specify filename(s)
Command format: DISKCOMP d: d: [/1] [/8]

DISKCOMP. You specified an incorrect switch or gave a filename in addition to a drive name.

Do you see the leftmost 0? (Y/N)

MODE. MODE displays this message to help you align the test pattern on your screen.

Do you see the rightmost 9? (Y/N)

MODE. MODE displays this message to help you align the test pattern on your screen.

Drive letter must be specified

FORMAT. You must specify the drive that you want to format.

Drive types or diskette types not compatible

DISKCOMP. You cannot compare a single-sided disk with a double-sided disk, or a high-density disk with a low-density disk. You should use FC to compare individual files on the disks.

Duplicate file name or File not found

CHKDSK, MS-DOS. You have tried to rename a file to a filename that already exists or the name you specified could not be found.

End of input file

EDLIN. The entire file was read into memory. If the file is read in sections, this message indicates the last section of the file is in memory.

Enter current Volume Label for drive *x:*

FORMAT. FORMAT asks you to enter the current volume label for verification before it formats the hard disk in the specified drive.

Enter new date:

MS-DOS. Either press Return, or enter the correct date and press Return. The format for the date is m-d-y, where *m* is a number from 1 to 12 indicating the month, *d* is a number from 1 to 31 indicating the current date, and *y* is number from 80 to 99 (or 1980 to 2099) indicating the year. The result of entering the date August 25th, 1986, is shown below.

```
Enter new date: 8-25-86
```

Enter new time:

MS-DOS. Either press Return, or enter the correct time and press Return. The format for the time is hh:mm:ss, where *hh* is the hour, *mm* is the minute, and *ss* is the second. Enter the hour in 24-hour format; that is, add 12 to the hours between 12:00 noon and midnight; enter either 00 or 24 for midnight. The results of entering 10:30 a.m. and 10:30 p.m respectively are shown below.

```
Enter new time: 10:30
Enter new time: 22:30
```

Entry error

EDLIN. The last command you typed contained a syntax error. Retype the command with the correct syntax and press Return.

Entry has a bad attribute (or link or size)

CHKDSK. This message may be preceded by one or two periods which indicate which subdirectory is invalid. If you have specified the /F switch, CHKDSK will attempt to correct the error.

Error in .EXE file

MS-DOS. The .EXE file you have asked MS-DOS to load has an invalid internal format.

Error reading/writing partition table

FORMAT. FORMAT could not read or write the partition table. You could run FDISK on the disk and then try formatting it again.

Errors found, F parameter not specified
Corrections will not be written to disk

CHKDSK. CHKDSK found errors on the disk. If you have not specified the /F switch, CHKDSK continutes printing messages, but will not correct the errors.

Errors on list device indicate that it
may be off-line. Please check it.

PRINT. Check to make sure that your printer is turned on.

Error writing to device

MS-DOS. You tried to send too much data to a device. MS-DOS was unable to write the data to the specified device.

EXEC failure

MS-DOS. MS-DOS either found an error when reading a command or the FILES command in the CONFIG.SYS file is set too low. Increase the value and restart MS-DOS.

FCB unavailable reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

FCB unavailable writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

fc: cannot open *filename* —No such file or directory

FC. One of the files that you specified doesn't exist. Check the directory for the correct filename.

fc: *filename* longer than *filename*

FC. After reaching the end of one of the files in a file comparison, the other file still has uncompared data left.

fc: incompatible switches

FC. You have specified switches that are not compatible (for example, /B and /L). You should not combine binary and ASCII comparison switches.

fc: out of memory

FC. You do not have enough memory to perform the comparison.

fc: no differences encountered

FC. The files are the same.

File allocation table bad or
File allocation table bad drive *x:*

MS-DOS, CHKDSK, or PRINT. The diskette or other storage media may be defective. Run CHKDSK, using the /F option, to repair any directory errors that may exist.

File cannot be copied onto itself

COPY, REPLACE, XCOPY. The source filename you specified is the same as the destination filename. Example: COPY A A.

File cannot be converted

EXE2BIN. The input file is not in the correct format.

File creation error

MS-DOS, XCOPY. You tried to add a new filename or replace a file that already exists in the directory. If the file already exists, it is a read-only file and cannot be replaced. Run CHKDSK on the disk to determine the cause of the error.

File is READ-ONLY

EDLIN. You cannot change this file because it is designated read-only.

Filename must be specified.

EDLIN. You did not specify a filename when you started EDLIN.

File not found

EDLIN, MS-DOS, PRINT, RECOVER. The filename you specified could not be found. Check to see that the pathname is accurate and that the file exists in the directory you specified.

FIND: Access denied

FIND. You cannot access the file. Make sure that the disk is not write-protected.

FIND: File not found

FIND. You specified a file that doesn't exist. Make sure you have typed the filename correctly.

FIND: Invalid number of parameters

FIND. You specified too many, or not enough options in the FIND command.

FIND: Invalid Parameter

FIND. One of the switches that you specified is wrong.

FIND: Read error in *filename*

FIND. The programs could not read the specified file.

FIND: Syntax error

FIND. Check to make sure that you have typed the command correctly.

First cluster number is invalid, entry truncated

CHKDSK. The file directory entry contains an invalid pointer to the data area. If you specified the /F switch, the file is truncated to a zero-length file.

First diskette bad or incompatible

DISKCOMP. Diskcomp cannot recognize the format on the source disk. You should run CHKDSK to help you identify the problem.

Fixups needed—base segment *hex:*

EXE2BIN. The source (.EXE) file contained information indicating that a load segment is required for the file. Specify the absolute segment address at which the finished module is to be located.

FOR cannot be nested

MS-DOS. Nesting of FOR statements is not allowed in a batch file.

Format another (Y/N)?

FORMAT. Type Y (for Yes) to format another disk.

Type N (for No) if you do not want to format another disk. If you accidentally type Y, you can abort the format process by typing Ctrl-C in response to the "Strike any key to begin formatting" message.

Format failure

FORMAT. MS-DOS could not format the disk. This message is always displayed in conjunction with an explanation as to why MS-DOS could not format the disk.

Formatting while copying

DISKCOPY. If the target diskette is not formatted the same as the source diskette, DOS automatically reformats it before the copy procedure is performed.

Format not supported on drive *x:*

FORMAT. You cannot use FORMAT to format the specified drive.

General failure reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

General failure writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Graphics characters loaded

GRAFTABL. The table of graphics has been loaded into memory.

Graphics characters already loaded

GRAFTABL. The table of graphics characters has already been loaded into memory.

Illegal device name

MODE. An invalid device was specified.

Incompatible system size

SYS. The system files IO.SYS and MSDOS.SYS occupy more space on the source disk than is available on the destination disk.

Incorrect DOS version

ATTRIB, CHKDSK, DISKCOMP, DISKCOPY, EDLIN, FORMAT, GRAFTABL, MODE, MORE, PRINT, RECOVER, REDIR, SHARE, SYS, SUBST, TREE, XCOPY. This message appears when attempting to run certain MS-DOS utilities that will not run under earlier versions of MS-DOS. The commands run only under the exact version of MS-DOS for which they were configured.

Incorrect number of parameters

JOIN, SUBST. You specified too many or too few options in the command line.

Incorrect parameter

ASSIGN, REDIR, SHARE. One of the options you specified is wrong.

Infinite retry on parallel printer timeout

MODE. Your printer is probably off-line or not ready. If the printer appears to be ready, you should restart your computer.

Initialization error

PCSLAVE. An error occurred that prevented the initialization of the COM1: serial port. Ensure that any serial cards are properly installed in the PC slave station.

Insert destination disk in drive *x:* and strike any key when ready

SYS. This message appears when you are using SYS to transfer the operating system with a single diskette drive. You should insert a disk in the appropriate drive and press any character or number key to begin processing.

Insert diskette for drive *x:* and strike any key when ready

MS-DOS. This message appears when MS-DOS is copying files. You should insert a disk in the appropriate drive and press any alphanumeric key to begin processing.

Insert diskette with batch file and press any key when ready

MS-DOS. You no longer have the disk containing the batch file you specified in the drive you originally specified. Reinsert the disk that contains the batch file in the appropriate drive.

Insert DOS diskette in drive *x:* and strike any key when ready

FORMAT. You have specified FORMAT /S but the disk in the default drive does not contain MS-DOS system files. Insert a disk with the files IO.SYS and MSDOS.SYS (bootable disk) in the drive specified.

Insert FIRST diskette in drive *x:*

DISKCOMP. DISKCOMP displays this message to prompt you to insert the first disk into the specified drive.

Insert new diskette for drive *x:* and strike any key when ready

FORMAT. Insert a blank disk into the appropriate drive and press any alphanumeric key to begin formatting. If there is any data on the disk, it will be destroyed by the format process.

Insert SECOND diskette in drive *x:*

DISKCOMP. DISKCOMP displays this message to prompt you to insert the second disk that you want to compare into the specified drive.

Insert source diskette into drive *x:*

DISKCOPY. Insert the disk to be copied into the specified drive.

Insert system diskette in drive *x:* and strike any key when ready

SYS. SYS needs a bootable disk from which to read the IO.SYS and MSDOS.SYS files. Insert a bootable disk into the specified drive and press any alphanumeric key to start the system copy process.

Insert target diskette into drive *x:*

DISKCOPY. You are running DISKCOPY and your source and destination drives are the same. Reinsert the destination disk into the specified drive.

Insufficient disk space

MS-DOS, REPLACE. The disk is full. It does not contain enough room to perform the specified operation.

Insufficient memory

CHKDSK, DISKCOMP, EDLIN, PCSLAVE, REPLACE. There is not enough memory to perform the specified operation.

Insufficient memory for system transfer

FORMAT. Your memory configuration is insufficient to transfer the MS-DOS system files IO.SYS and MSDOS.SYS (the /S switch).

Insufficient room in root directory.
Erase files in root and repeat CHKDSK

CHKDSK. CHKDSK always recovers lost files into the root directory. In this case, your root directory is full. Delete some files in your root directory to make room for the lost files.

Intermediate file error during pipe

MS-DOS. The pipe operation makes use of temporary files on the disk which will automatically be deleted after the piping process is complete. An error has occurred in one of these files.

Invalid baud rate specified

MODE. Valid baud rates are 110, 150, 300, 600, 1200, 2400, 4800, or 9600 where the first two characters are required and the rest are ignored.

Invalid characters in volume label

FORMAT. The volume label should contain only up to 11 alphanumeric characters.

Invalid COMMAND.COM
Insert COMMAND.COM disk in default drive
and strike any key when ready

MS-DOS. The application you have just run used up almost all of memory. It is necessary for MS-DOS to reload the COMMAND.COM file from disk. However, MS-DOS cannot find COMMAND.COM on the disk or the copy found is invalid. Insert a disk into the default drive which contains a copy of COMMAND.COM similar to the version on the disk with which you started MS-DOS.

Invalid country code

MS-DOS. You have specified a country number in your CONFIG.SYS file which is not in the table of files configured in this implementation of MS-DOS. Consult the documentation on CONFIG.SYS for more information on supported countries.

Invalid date

BACKUP, DATE, RESTORE, XCOPY. You specified an invalid date. Specify the date correctly and try the command again.

Invalid device

MS-DOS. The device specified was not CON, NUL, AUX, or PRN.

Invalid device parameters from device driver

FORMAT. FORMAT displays this message when the number of hidden sectors is not evenly divisible by the number of sectors per track (i.e., the partition does not start on a track boundary). This might happen if you tried to format a hard disk that previously had been formatted with MS-DOS 2.x without first running FDISK.

Invalid directory

MS-DOS. The directory you specified either does not exist or is invalid. Check to see that you entered the directory name correctly.

Invalid disk change reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Invalid disk change writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Invalid drive in search path

MS-DOS. The drive does not exist.

Invalid drive or file name

EDLIN, RECOVER. Either the filename or the device letter or name is incorrect. Specify a valid drive or a valid filename.

Invalid drive specification

CHKDSK, DISKCOMP, DISKCOPY, FORMAT, PRINT, REPLACE, SYS, TREE, XCOPY. The device letter or name you specified doesn't exist. Specify a valid letter or name.

Invalid environment size specified

COMMAND. You gave an invalid number of bytes with the /E switch. You must specify a number between 128 and 32768.

Invalid number of parameters

MS-DOS, ATTRIB, RECOVER, XCOPY. You have specified the wrong number of options in the command line.

Invalid parameter

CHKDSK, DISKCOMP, DISKCOPY, EDLIN, FORMAT, MODE, PRINT, REPLACE, SORT, SUBST, SYS, TREE, XCOPY. One of the options that you have specified is wrong.

Invalid path or file name

COPY, XCOPY. Specify a valid pathname or filename.

Invalid path, not directory, or directory not empty

MS-DOS. You are unable to remove the directory requested for one of the specified reasons.

Invalid sub-directory entry

CHKDSK. The subdirectory that you specified either does not exist or is invalid. Check to see that you entered the subdirectory name correctly.

Invalid time

BACKUP, RESTORE, TIME, XCOPY. You specified an invalid time.

Invalid volume ID

FORMAT. You entered a volume label that doesn't match the label on the hard disk you want to format. The format process is ended.

Invalid working directory

MS-DOS. Your disk is bad. Replace the disk or make another copy from your backup system disk.

Label not found

MS-DOS. There is a GOTO command to a nonexistent label in a batch file.

Line too long

EDLIN. During a REPLACE command, the string given as the replacement caused the line to expand beyond 253 characters. Divide the long line into two lines and retry the REPLACE command.

List output is not assigned to a device

PRINT. When you first run print, it asks you what device you want to specify as a print spooler. This message appears if PRINT is set up for a nonexistent device.

Lock violation reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Lock violation writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

LPT#: not redirected

MODE. MODE could not redirect the parallel printer port. Check to see that you have specified the proper options.

LPT#: redirected to COM#:

MODE. Output to the parallel printer port will now be sent to the specified asynchronous communications port.

Memory allocation error. Cannot load MS-DOS, system halted

MS-DOS. Restart MS-DOS. If this error persists, make a new copy of the MS-DOS disk from your backup copy of the system disk.

—More—

MORE. Press the space bar to view more of the file or directory.

MORE: Incorrect DOS version

MORE. MORE will not run on versions of MS-DOS previous to 2.0.

Must specify destination line number

EDLIN. You must specify a destination line number when you are copying and inserting lines with EDLIN.

Must specify ON or OFF

MS-DOS. The command requires either an ON or an OFF argument.

Name of list device [PRN]:

PRINT. This prompt appears when PRINT is run the first time. Any valid device may be specified and that device then becomes the PRINT output device.

Network has not been started

REDIR. You must start the network before you can use the REDIR command.

New file

EDLIN. This message is printed if EDLIN does not find a file with the name you specified. If you are creating a new file, ignore this message. If you do not intend to create a new file, check to see that you correctly typed the filename of the file you wish to edit.

No files added/replaced

REPLACE. The REPLACE command did not add or replace any files.

No file found *filename*

REPLACE. REPLACE could not find matching source or target files.

No files match *filespec*

PRINT. A filespec was given for files to add to the queue, but no files match the specification.

No free file handles
Cannot start COMMAND.COM, exiting

MS-DOS. Restart MS-DOS. If this message persists, increase the FILES parameter in the CONFIG.SYS file.

Non-DOS disk error reading drive *x:*

MS-DOS, PRINT. Device error. Take the action described under the message Abort, Retry, Ignore?

Non-system disk or disk error
Replace and strike any key when ready

FORMAT, SYS. Place the proper disk in the disk drive and press any alphanumeric key to continue.

No paper error writing device *dev*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

No path

MS-DOS. You have typed PATH to find out what your search path is. There is no current command search path.

No room for system on destination disk

SYS. There is not enough room for the system files on the destination disk. Delete some files to make room for the system files or use another disk. You may need to reformat the disk to put the system on it.

No room in directory for file

EDLIN. You have tried to save a file to the root directory but it is full. Subdirectories are not limited in size as is the root directory.

No subdirectories exist

TREE. The disk in the drive you specified does not contain subdirectories.

No system on default drive

SYS. The system files do not exist on the disk in the default drive.

Not enough memory

JOIN, SHARE, SUBST. There is not enough memory for MS-DOS to run the specified command.

Not enough room to merge the entire file

EDLIN. There was not enough room in memory to hold the file during a TRANSFER command. You must free some memory by writing some files to disk or deleting some files before you can transfer this file.

Not found

EDLIN. You have specified a SEARCH or a REPLACE command that was unable to find a further occurrence of the specified Search or Replace string.

Not ready error reading drive *x:*

MS-DOS. Device error. (*x* is a letter indicating the storage device.) Take the action described under the message Abort, Retry, Ignore?

PCMASTER. You issued a command for a PC file transfer. Make sure the diskette is inserted in the proper drive and the latch is secured.

Make sure the diskette is in the drive that corresponds to the device letter you specified in the command. See the section Assignment of Device Letters in Chapter 11 to determine which device letters you should specify.

Not ready error writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

O.K.?

EDLIN. This prompt occurs during SEARCH and REPLACE command processing. If you press any key except Y or Return, the search or replace continues.

Out of environment space

MS-DOS. There is not enough room in the program environment to accept more data. You can use the COMMAND command with the /E switch to increase the size of the environment.

Parameters not compatible

FORMAT, REPLACE. You have specified switches that cannot be used together.

Parameters not compatible with fixed disk

FORMAT. You have used a switch that is not compatible with the specified drive.

Parameters not supported by Drive

FORMAT. The device driver for the specified drive does not support generic IOCTL function requests.

Path not found

CHKDSK, REPLACE, SUBST. You specified an invalid path.

Path too long

REPLACE, XCOPY. The pathname you specified was too long. You may have to change directories to replace files in deep subdirectories.

Press any key to begin adding files

REPLACE. When you specify the /W switch, REPLACE displays this message to prompt you to start replacing files.

PCMaster has been installed
The first slave device will be drive x:

PCMASTER. The PCMASTER device driver has been installed in the system. When you run the PC File Transfer utilities, the first device at the slave station will be assigned drive letter *x:* .

Press any key to begin recovery of the file(s) on drive *x:*

RECOVER. This prompt is issued before you recover a disk or file. Press any alphanumeric key to begin the recover process. If you wish to discontinue this operation, press Ctrl-C.

Press any key when ready

DISKCOMP, DISKCOPY. This prompt occurs when you are copying disks. When you have inserted the disks into the appropriate drives, press any alphanumeric key to begin the DISKCOPY process. If you wish to discontinue this operation, press Ctrl-C.

PRINT queue is empty

PRINT. There are no files waiting to be printed.

PRINT queue is full

PRINT. There is room for 10 files in the list of files waiting to be printed.

Probable non-DOS disk
Continue (Y/N)?

CHKDSK. The disk you are using is not recognized by this version of MS-DOS. The disk either was created by another system with a format that is not supported on this version of MS-DOS or is not an MS-DOS disk. Do not continue processing if CHKDSK has returned this message for a removable disk. If this message is returned for a hard disk, the information describing the characteristics of the disk to MS-DOS has been destroyed. In this case, you may continue CHKDSK processing.

Processing cannot continue

CHKDSK. There is not enough memory in your machine to process CHKDSK for this disk. You must obtain more memory to run CHKDSK.

Program too big to fit in memory

MS-DOS. You must acquire more memory to run your application. It is possible that some applications you have run are still using some memory. You may try to restart MS-DOS; however, if you still receive this message, you must acquire more memory.

Read error in *filename*

FIND. MS-DOS could not read the file.

Read fault error reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Redirector already installed

REDIR. The Redirector can only be installed once.

Re-insert diskette for drive *x:*

FORMAT. Reinsert the disk being formatted in the indicated drive.

Resident part of PRINT installed

PRINT. This is the first message that MS-DOS displays when you issue the PRINT command. It means that available memory has been reduced by several thousand bytes to process the PRINT command concurrent with other processes.

Resident portion of MODE loaded

MODE. Part of the MODE program is now resident in memory.

Resynch failed. Files are too different.

FC. The number of lines in the internal line buffer is less than the number of consecutive, differing lines. You should specify the /LB switch with a larger number if you want to display all of the file differences.

SECOND diskette bad or incompatible

DISKCOMP. The second disk does not contain the same format as the first disk, or DISKCOMP does not recognize the format of the second disk. You should run CHKDSK to help you identify the problem.

Sector not found error reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Sector not found error writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Sector size too large in file *filename*

MS-DOS. The specified device driver loaded by CONFIG.SYS uses a sector size larger than that of any other device driver on the system. You cannot run this device driver.

Seek error reading drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Seek error writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Serial port specified does not exist on system

PCSLAVE. Either the serial port you specified doesn't exist, or your computer does not have a serial port. To use the COM1 serial port, enter one of the following commands:

```
PCSLAVE
or
PCSLAVE 1
```

To use the COM2 serial port, enter the following command:

```
PCSLAVE 2
```

SHARE already installed

SHARE. Share can only be installed once.

Sharing violation reading drive *x:*

SHARE. Device error. Take the action described under the message Abort, Retry, Ignore?

Slave devices are not usable
PCMaster requires DOS version 2.10 or higher

PCMASTER. The PCMASTER device driver has been installed in the system, but the slave devices can't be used. This version of PCMASTER requires that you use MS-DOS version 2.10 or higher.

SORT: Incorrect DOS version

SORT. Sort will not run on MS-DOS versions prior to MS-DOS 2.0.

SORT: Insufficient disk space

SORT. The disk is full.

SORT: Insufficient memory

SORT. There is not enough memory to run the SORT program.

Source and target diskettes are not the same format.
Cannot do the copy

DISKCOPY. You must have the same size and kind of disks to run DISKCOPY. Example: you cannot copy from a single-sided disk to a double-sided disk. Reformat the target disk to be of the same type as the source disk. Use XCOPY when copying from 5¼-inch disks to 3½-inch disks.

Source path required

REPLACE. You did not specify a source path.

Specified drive does not exist, or is non-removable

DISKCOMP. You must specify the name of a valid floppy drive. DISKCOMP cannot compare hard disks.

Specified MS-DOS search directory bad

MS-DOS. The SHELL statement in the CONFIG.SYS file is incorrect. The place that you have told MS-DOS to find COMMAND.COM does not exist, or COMMAND.COM is not in that place.

Strike a key when ready . . .

MS-DOS. This prompt occurs during command processing and is always accompanied by another message. This message is also displayed if you have inserted a PAUSE command in a batch file. Usually, you are asked to insert disks into appropriate drives before this prompt. Press any alphanumeric key to begin command processing.

Syntax error

MS-DOS, FIND. Check to make sure that you have typed the command correctly.

System transferred

FORMAT, SYS. The system files MSDOS.SYS and IO.SYS have been transferred during FORMAT or SYS command processing.

Terminate batch job (Y/N)?

MS-DOS. If you press Ctrl-C while in batch mode, MS-DOS asks you whether or not you wish to end batch processing. Press Y to end processing. Press N to continue the batch job.

Too many files open

EDLIN. MS-DOS could not open the file to edit on the .BAK file due to lack of system file handles. Increase the value of the FILES command in the CONFIG.SYS file.

Track 0 bad—disk unusable

FORMAT. FORMAT can accommodate for defective sectors on the disk except for those near the beginning. In this case, use another disk.

Unable to create a directory

MS-DOS, XCOPY. MS-DOS could not create the directory you specified. Check to see that there is not a name conflict (you may have a file by the same name) or the disk may be full.

Unable to shift Screen

MODE. MODE is unable to shift the test pattern on the screen any farther.

Unexpected DOS Error *nnn*

REPLACE. An unexpected error occurred, where *nnn* is the MS-DOS error number.

Unrecognized command in CONFIG.SYS

MS-DOS. There is an invalid command in your CONFIG.SYS file.

Unrecoverable error in directory
Convert directory to file (Y/N)?

CHKDSK. If you respond Y to this prompt, CHKDSK will convert the bad directory into a file. You can then fix the directory yourself or delete it.

usage: fc [/a] [/b] [/c] [/1] [/1b n] [/w] [/t] [/n] [/NNNN] file1 file2

FC. One of the switches that you specified is invalid.

VERIFY is off (or on)

MS-DOS. This message tells you the current setting of VERIFY.

Volume in drive *x:* has no label

MS-DOS, LABEL. This is an informational message displayed in response to the DIR command.

Volume in drive *x:* is *label*

MS-DOS, LABEL. This is an informational message displayed in response to the DIR command.

Volume label (11 characters, ENTER for none)?

FORMAT, LABEL. This message is displayed when you specify the /V switch in the FORMAT command, or when you give the LABEL command. Specify a volume label or press Return to indicate that you do not want a volume label for this disk.

WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE X WILL BE LOST!
Proceed with Format (Y/N)?

FORMAT. There is data on the hard disk that you are trying to format. If you want to format the disk and destroy the data, press Y (for Yes). If you do not want the files on your hard disk erased, press N (for No). Copy the files to a floppy disk and repeat the FORMAT command.

Warning—directory full

RECOVER. The root directory is too full for RECOVER processing. Delete some files in the root directory to free space.

Warning: Read error in EXE file

EXE2BIN. The amount read was less than the size of the header. This is a warning message only.

Write fault error writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Write protect error writing drive *x:*

MS-DOS. Device error. Take the action described under the message Abort, Retry, Ignore?

Appendix A: ASCII Escape Code Sequences

This chapter consists of Table A-1. It lists the ANSI standard ASCII escape code sequences the GRiDCase supports. To use these sequences, you must have the ANSI.SYS file and the DEVICE = ANSI.SYS command in your CONFIG.SYS file (see Chapter 5).

The abbreviations used in Table A-1 are explained below:

Code Sequence Item	Code Equivalent
ESC	Hexadecimal 1B or decimal 27, the ASCII escape character.
#	ASCII numeric character(s) that represents the number of rows or columns.
col	ASCII numeric character(s) that represents a column position on the screen; valid values are 1 (far left character position) through 80 (far right character position).
row	ASCII numeric characters that designate a row position on the screen; valid values are 1 (top line of screen) through 26 (bottom line of screen).

NOTE: If #, col, or row are not specified, then zero (0) is assumed unless otherwise stated. The cursor stops at the top, left, right, or bottom margin if the value specified for #, col, or row specifies a position beyond that margin.

Table A-1. *The GRiDCase Cursor and Screen Control Sequences*

Function	Control Sequence
Cursor Position Save	Sequence: ESC [s Hexadecimal: 1B 5B 73 Current cursor position is saved.
Cursor Position Restore	Sequence: ESC [u Hexadecimal: 1B 5B 75 Cursor is restored to previously saved position.
Delete Lines	Sequence: ESC [# M Hexadecimal: 1B 5B # 4D Deletes one or more lines following the current cursor position, where # is the number of lines to be deleted. If you don't specify #, one line is deleted. After deletion, the cursor remains stationary and the lines below the deleted line(s) scroll up.
Erase Screen	Sequence: ESC [2 J Hexadecimal: 1B 5B 32 4A Erase screen and move cursor to home position.
Erase To End of Line	Sequence: ESC [0 K Hexadecimal: 1B 5B 30 4B Erase to end of line including cursor position.
Erase To End of Display	Sequence: ESC [0 J Hexadecimal: 1B 5B 30 4A Erase to end of display.
Insert Lines	Sequence: ESC [# L Hexadecimal: 1B 5B # 4C Inserts one or more lines following the current cursor position, where # is the number of lines to be inserted. If you don't specify #, one line is inserted. After insertion, the cursor remains stationary and the lines below the inserted line(s) scroll down.
Reverse Video Off	Sequence: ESC [0 m Hexadecimal: 1B 5B 30 6D Turn reverse video off.

Function	Control Sequence
Reverse Video On	Sequence: ESC [7 m Hexadecimal: 1B 5B 376D Turn reverse video on.
Move Cursor Down	Sequence: ESC [# B Hexadecimal: 1B 5B # 42 Cursor moved down # lines.
Move Cursor Right	Sequence: ESC [# C Hexadecimal: 1B5B # 43 Cursor moved right # lines.
Move Cursor Left	Sequence: ESC [# D Hexadecimal: 1B 5B # 44 Cursor moved left # lines.
Move Cursor To Location	Sequence: ESC [row;col H ESC [row;col f Hexadecimal: 1B 5B row 3A col 48 1B 5B row 3A col 66 Cursor moved to the location specified by *row* and *col*.
Move Cursor Up	Sequence: ESC [# A Hexadecimal: 1B5B # 41 Cursor moved up # lines.
Reassign Keyboard	Sequence: ESC [Pn ; Pn ; # Pn p ESC ["string" ; p ESC [Pn ; "string" ; Pn ; Pn ; "string" ; Pn p Any other combination of strings and decimal numbers. Hexadecimal: 1B 5B 50 6E 3B 50 6E # 3B 50 70 1B 5B string 3B 70 1B 5B string 3B 50 6E 3B 50 6E

Function	Control Sequence
Reassign	The method of reassigning keys on the GRiDCase is the same as that on the IBM PC or PC/XT, as shown in the examples below.
	1. In an escape sequence, the second ASCII code for a key redefines the previous ASCII code specified. For example, the following sequences reassign the Q and q key to the A and a key (and vice versa): ESC [65 ; 81 p A becomes Q ESC [97 ; 113 p a becomes q ESC [81 ; 65 p Q becomes A ESC [113 ; 97 p q becomes a **NOTE:** If the first code in the sequence is zero (NUL) then the first and second code become an extended ASCII redefinition.
	2. The following statement reassigns the F10 key to a DIR command followed by a carriage return: ESC [0 ; 68 ; “dir” ; 13 p The 0;68 is the extended ASCII code for the F10 key. Decimal 13 is a carriage return.
Reset Mode (RM)	Sequences: ESC [= Ps l ESC [= l ESC [= 0 l ESC [? 7 l Hexadecimal: 1B 5B 3D 50 73 6C 1B 5B 3D 6C 1B 5B 3D 30 6C 1B 5B 3F 37 6C Parameters are the same as SM (Set Mode) except parameter 7 resets wrap at end of line mode.

Function	Control Sequence	
Set Graphics Rendition (SGR)	Sequence:	ESC [Ps ; # ; Ps m
	Hexadecimal:	1B 5B 50 73 # 50 73 6D
	Invokes the graphic rendition specified by the parameter #, which is a value listed below. All characters are rendered according to the function specified by # until the next occurrence of SGR.	
	Parameter	**Parameter Function**
	0	All Attributes Off
	1	Bold On
	4	Underscore On (monochrome displays only).
	5	Blink On
	7	Reverse Video On
	8	Concealed On*
	30	Black foreground*
	31	Red foreground*
	32	Green foreground*
	33	Yellow foreground*
	34	Blue foreground*
	35	Magenta foreground*
	36	Cyan foreground*
	37	White foregound*
	40	Black background*
	41	Red background*
	42	Green background*
	43	Yellow background*
	44	Blue background*
	45	Magenta background*
	46	Cyan background*
	47	White backgound*
	*ISO 6429 standard	
Status Information	Sequence:	ESC [6 n
	Hexadecimal:	1B 5B 36 6E
	Returns information on the current cursor position. This is in the keyboard queue in the following format:	
	ESC [row; col R	

Index

GRiD Publications Customer Response Card

GRiD Systems wants to supply you with the finest possible guides and manuals. Therefore, your feedback is important. To help us maintain the highest quality, please use this form to comment on the usefulness and readability of this publication. Suggest additions and deletions, and list specific problems, if any. Be sure to include page numbers where applicable.

MS-DOS Reference: Release 3.2 August 1986

Thank you for your cooperation!

GRiD Systems Corporation 47211 Lakeview Blvd., P.O. Box 5003
Fremont, CA 94538

tape or staple here

fold here

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 38 FREMONT, CA U.S.A.

POSTAGE WILL BE PAID BY ADDRESSEE

GRiD Systems Corporation
Attention: Publications Manager
47211 Lakeview Blvd.
P.O. Box 5003
Fremont, CA 94538